JPN BUILD VERSION 0.96

System Operating Manual
for
e:cue *media engine*
Product Family
製品取扱説明書

1. はじめに 10
1.1. E:CUE プログラムの概要 10
2. 安全にお使いいただくために 15
3. ショートガイド／クイックスタート 16
3.1. DMX アウトプットディバイスのアライメント 16
3.2. E:CUE フェーダーユニットのアライメント 17
3.3. DMX ターミナルの定義エラー! ブックマークが定義されていません。
3.4. シーンを作成する 16
3.5. シーンの保存 17
3.6. 保存したシーンを呼び出す 21
3.7. ウィザード 21
3.7.1. チェイサーウィザード 21
3.7.2. スクロールウィザード 19
3.7.3. チェックイットウィザード 20
4. PROGRAMMER のメニュー構成 24
5. DMX アウトプットディバイスのアライメント 32
5.1. アウトプットディバイスを定義する 32
5.2. モニターステータス 34
5.3. キューリストを出力デバイスにエクスポート ... 34
5.4. NANO（ナノ） 34
5.5. TWILIGHT（トワイライト） 36
5.6. E:NODE（イーノード） 36
6. DMX ターミナルの定義 38
6.1. デバイスを追加する 36
6.2. デバイスの特別設定（オプション） 40
6.3. セクションを使う 38

7. PROGRAMMER を使ってみよう 42

7.1. ナビゲーター 39
7.2. PROGRAMMER の 3 つのビューモード 43
7.2.1. Programmer ビュー 43
7.2.2. キューリストコンテント 44
7.2.3. ステージアウトプット 41
7.3. 新しいシーンのためにデバイスを選択する 46
7.4. コマンドラインエントリー 49
7.5. ハイライト機能 49
7.6. ブラインド機能 49
7.7. フリーズ機能 47
7.8. カラーピッカー 47
7.9. ライブ FX エフェクトジェネレーター 51
7.10. プリセット機能 53
7.11. シーンを保存する（キューの登録） 53

8. キューとキューリストを使用する 55

8.1. 新しいキューをつくる 57
8.2. チェイサーウィザード 53
8.3. スクローラーウィザード 54
8.4. チェックイットウィザード 56
8.5. キューを変更する 57
8.6. キューのプロパティ 57
8.7. キューリストのプロパティ 60
8.8. タイムコード 62
8.9. オートメーション 63
8.10. サウンド TO ライト 64

9. WYSIWYG を使って視覚化する 72

9.1. STEP 1: TCP/IP プロトコル 72
9.2. STEP 2: PROGRAMMER の設定 73
9.3. STEP 3: WYGINSTALL 74
9.4. STEP 4: WYSIWYG の設定 74

9.5. STEP 5: デバイスをパッチする 74
10. CAPTURE を使って視覚化する 75
11. 設定 69
11.1. ショープロパティ 69
11.2. オプション 72
12. オートメーション 73
12.1. トリガーマシン 81
12.2. マクロ 75
12.3. E:CUEPROGRAMMER について 87
12.3.1. Variables 87
12.3.2. int 87
12.3.3. int arrayname[size] 88
12.3.4. string 88
12.3.5. if 88
12.3.6. for (init; condition; increment) command; 89
12.3.7. return 89
12.3.8. switch, case, break 89
12.3.9. while 90
12.3.10. exit 91
12.3.11. Binary Operators 91
12.3.12. Assignment Operators 91
12.3.13. String Operators 91
12.3.14. Syntax 92
12.3.15. clear 92
12.3.16. printf 92
12.3.17. alert 92
12.3.18. abs 93
12.3.19. check 93
12.3.20. random 93

12.3.21. strlen 93
12.3.22. val 93
12.3.23. strcmp 93
12.3.24. stricmp 94
12.3.25. getchar 94
12.3.26. setchar 94
12.3.27. getuchar 94
12.3.28. setuchar 94
12.3.29. getarg 95
12.3.30. midstr 95
12.3.31. createstring 95
12.3.32. format 96
12.3.33. exists 96
12.3.34. undefined 96
12.3.35. size 96
12.3.36. GetTerminalCount 96
12.3.37. TerminalSend 96
12.3.38. TerminalSetup 97
12.3.39. GetCuelistStatus 97
12.3.40. GetCuelistName 97
12.3.41. GetMutexStatus 98
12.3.42. MutexRelease 98
12.3.43. StartCuelist 98
12.3.44. StopCuelist 98
12.3.45. GetCurrentCue 98
12.3.46. ReleaseAll 99
12.3.47. PowerUp 99
12.3.48. PowerDown 99
12.3.49. MediaPlay 99
12.3.50. MediaStop 99
12.3.51. MediaEject 100
12.3.52. SetOutput 100
12.3.53. ClrOutput 100

12.3.54. GetInput *100*
12.3.55. MidiNoteOn *100*
12.3.56. MidiNoteOff *101*
12.3.57. SendMidi *101*
12.3.58. TimecodeStart *101*
12.3.59. TimecodeStop *101*
12.3.60. SendString *102*
12.3.61. SendBinary *102*
12.3.62. GetGrandMaster *102*
12.3.63. SetGrandMaster *102*
12.3.64. GetSubMaster *102*
12.3.65. SetSubMaster *103*
12.3.66. GetSpeedMaster *103*
12.3.67. SetSpeedMaster *103*
12.3.68. SetSectionRights *103*
12.3.69. GetSectionRights *104*
12.3.70. Call *104*
12.3.71. NiceTime *104*
12.3.72. CopyCuelist *105*
12.3.73. PasteCuelist *105*
12.3.74. SetMutexID *105*

13. フェーダーユニット 93

13.1. フェーダーユニットのアライメント 93

14. メディアエンジン 95

14.1. 概要 95
14.2. メディアエンジンのメインメニュー 108
14.2.1. サブメニュー再生 *96*
14.2.2. サブメニューログ *96*
14.3. メディアエンジンのシステムメニュー 111
14.3.1. DMX／モニター *111*
14.3.2. DMX／テスト *111*

14.3.3. PLC／ビュー 112
14.3.4. サウンド／再生 99
14.3.5. サウンド／停止 99
14.3.6. メディア／再生 99
14.3.7. メディア／停止 100
14.3.8. Midi／ビュー 100
14.3.9. SMPTE／ビュー 100
14.3.10. オプション／Misc 100
14.3.11. オプション／Net 100
14.3.12. オプション／ゲート 101
14.3.13. 時計／設定 102
14.3.14. フロッピー／ロード 102
14.3.15. フロッピー／保存 102
14.3.16. フロッピー／アップデート 102
14.3.17. Misc／アバウト 115
14.3.18. Misc／キー 115
14.3.19. ログイン／ログイン 102
14.3.20. ログイン／パスワード変更 102
14.4. E:CUE フェーダーユニットのアライメント .. 116
14.5. E:CUE キーパッドのアライメント 116

15. 追記 104

15.1. ライブラリーエディタ 104
15.1.1. プロデューサーの設定 118
15.1.2. DMX ターミナル（灯体）の設定 118
15.1.3. チャンネルの設定 118
15.1.4. レンジ 107
15.2. キーボードのコンフィギュレーション 120
15.3. デュアルヘッド 120

16. テクニカルデータ 122

16.1. E:CUEPROGRAMMER ソフトウェア 122
16.1.1. 概要 109

16.2. フェーダーユニット 125
16.2.1. 概要 111
16.2.2. RS232 インターフェイス 126
16.2.3. Midi インターフェイス 126
16.2.4. パワー 112
16.3. メディアエンジン 127
16.3.1. 概要 113
16.3.2. メディアエンジンインターフェイス 129
16.3.3. オプション 100
16.3.4. SMPTE インターフェイス（6.5mm プラグ) 131
16.3.5. Midi インターフェイス (5 ピン DIN プラグ) 131
16.3.6. DMX インターフェイス (5 ピン XLR プラグメス) 131
16.3.7. RS232 インターフェイス (9 ピン Sub-D プラグオス) 131
16.3.8. SPS インターフェイス (Phoenix プラグ、10 ピンオス) 132
16.3.9. オーディオ出力 (3 ピン XLR プラグオス) 132
16.4. NANOエラー! ブックマークが定義されていません。
16.4.1. RS232 (ピン Sub-D プラグオス) 133
16.4.2. 電源部 (2.1mm プラグオス) 133
16.4.3. DMX インターフェイス (5 ピン XLR プラグメス) 133
16.5. TWILIGHT 119
16.5.1. 概要 119
16.5.2. DMX インターフェイス (5 ピン XLR プラグメス) 135
16.5.3. USB 135

17. サポート 120

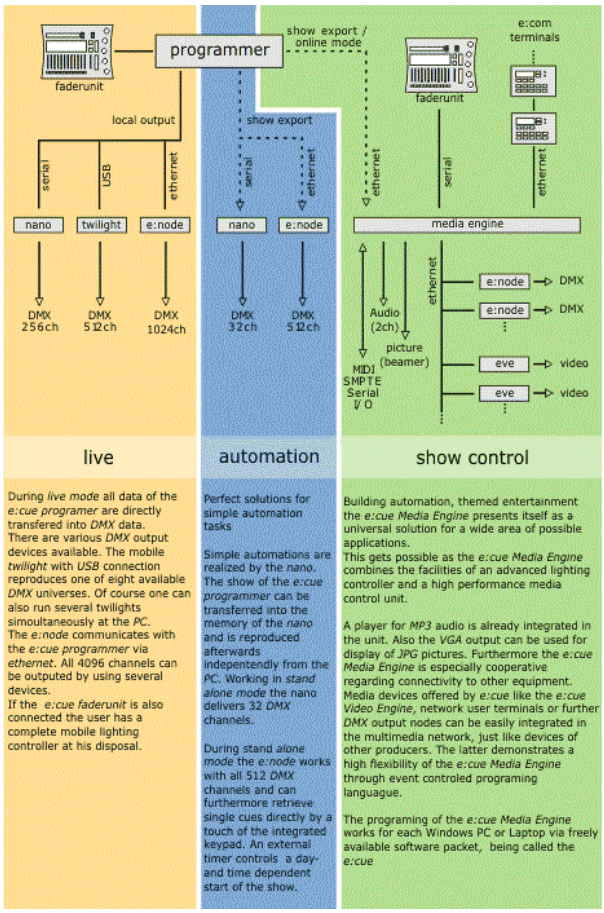
18. 著作権および保証に関する記述 137

19. 索引...エラー! ブックマークが定義されていません。

1. はじめに

1.1. e:cue Programmer の概要

E:cue programmer は、e:cue ファミリー製品の核となる製品で、複雑なライティングデザインに対し直感的なプログラミングとオペレーションを約束する先進のライティングソフトウェアです。
ライブステージでの使用だけではなく、e:cue ファミリーは建築照明（店舗照明などのインストールモデル）分野にも非常に力を入れています。
e:cue Media Engine（マルチメディアインターフェース）と TCP/IP でオンライン接続し、プログラムデータを伝送することで音楽、映像、照明の 3 元同期が実現するだけでなく、その強固な連係は固定設備におけるライティングデザインをより豊かなものにします。今までカスタムコンピューターでの外注プログラミングが必要であった建築照明も、Programmer のマクロ機能を使ってマクロ文を構成することにより、デザイナーの高い要求も難なくこなすことができます。
また、programmer は CAST 社の Wysiwyg などの 3D レンダリングソフトウェアとの完全互換を実現しています。フィールドワークをさらに広げ、もはやライティングデザインの場所をも選びません。
このように非常に高い機能と拡張性を持ち合わせたこのソフトウェアを存分にご堪能いただければ幸いです。


programmer
faderunit
local output
serial
USB
ethernet
nano
twilight
e:node
DMX 256ch
DMX 512ch
DMX 1024ch
show export / online mode
show export
serial
ethernet
nano
e:node
DMX 32ch
DMX 512ch
e:com terminals
faderunit
ethernet
serial
ethernet
media engine
ethernet
e:node
DMX
e:node
DMX
Audio (2ch)
picture (beamer)
MIDI SMPTE Serial I/O
eve
video
eve
video
live
automation
show control
During live mode all data of the e:cue programer are directly transfered into DMX data. There are various DMX output devices available. The mobile twilight with USB connection reproduces one of eight available DMX universes. Of course one can also run several twilights simoultaneously at the PC. The e:node communicates with the e:cue programmer via ethernet. All 4096 channels can be outputed by using several devices. If the e:cue faderunit is also connected the user has a complete mobile lighting controller at his disposal.
Perfect solutions for simple automation tasks
Simple automations are realized by the nano. The show of the e:cue programmer can be transferred into the memory of the nano and is reproduced afterwards indepentendly from the PC. Working in stand alone mode the nano delivers 32 DMX channels.
During stand alone mode the e:node works with all 512 DMX channels and can furthermore retrieve single cues directly by a touch of the integrated keypad. An external timer controls a day- and time dependent start of the show.
Building automation, themed entertainment the e:cue Media Engine presents itself as a universal solution for a wide area of possible applications. This gets possible as the e:cue Media Engine combines the facilities of an advanced lighting controller and a high performance media control unit.
A player for MP3 audio is already integrated in the unit. Also the VGA output can be used for display of JPG pictures. Furthermore the e:cue Media Engine is especially cooperative regarding connectivity to other equipment. Media devices offered by e:cue like the e:cue Video Engine, network user terminals or further DMX output nodes can be easily integrated in the multimedia network, just like devices of other producers. The latter demonstrates a high flexibility of the e:cue Media Engine through event controled programing languague.
The programing of the e:cue Media Engine works for each Windows PC or Laptop via freely available software packet, being called the e:cue

e:cue システム全体の概念図

Programmer 技術仕様

必要システム最小構成

ペンティアム「 CPU を内蔵した PC/AT 互換機（PC9800 シリーズを除く）
64MB RAM 必須
OS: Microsoft Windows 98SE/2000/NT/ME/XP ベースで動作(2002 年 10 月現在)

主な機能

- 非常に簡単に操作できるソフトパッチ部
- DMX コントロールライティングディバイスをフルサポート。
- 最大 4096DMX チャンネルをコントロール。
- マクロ機能搭載。
- 著明メーカーの灯体を網羅したスタンダードライブラリー
- 1 ディバイスで複数の灯体グループを一元化してコントロール
- ユーザー自身でオリジナル灯体データが作成できるライブラリエディターをバンドル
- 複数灯体のグルーピング機能
- パッチデータをはじめとするさまざまな情報を CSV 形式でエクスポート可能
- プログラミングセクション
- Programmer、キューリストコンテンツおよび DMX 出力の 3 つのビューを切替表示
- サークルやウェーブなどを簡単に作成できる機能豊かなエフェクトエンジン
- ハイライト／ブラインド／フリーズ機能搭載
- カラーピッカーによるスピーディーな色決め（ウォッシュ使用時）が可能。
- キューフェードにリニア／ソフトスタート／ソフトストップ／ソフトモーションを設定可能。
  また、任意のグループや列の中央／左右から徐々にスタートできるフェードディレイ機能を追加。
- 通常のキュー進行に加え、外部タイムコード／MIDI／ビートなどからのトリガーも可能。
- マクロ機能搭載。各キューもしくはキューリストに最大 5 つまでのアクション（音を出したり画像を出したり他のキューに連動させたり etc...）を登録可能。また、マクロはユーザー自身でプログラミングすることができ、C 言語を使ってさらに複雑なマクロも構成することが可能。（マクロ作成オプションも用意しています）
- チェイススピード学習機能
- テキストファイルに既存のショーデータ詳細をエクスポート可能
- デュアルスクリーンサポート

内蔵ウィザード

- チェイサーウィザード：キューリストからチェイスリストを自動的に作成します。
- チェックイットウィザード：接続された灯体各種をチェックするのに最適なウィザードです。

- スクロールテキストウィザード：マトリクス上に配置したパーライトなどを使って、文字がスクロールするキューリストを作成するウィザードです。

Lights Library（灯体ライブラリー）
さまざまな種類の灯体や他の DMX 照明機器が格納されているライブラリーです。著明メーカーから提供された各種データを元に作成しています。また、ここに存在しない機材についてはライブラリエディタを使って登録することも可能です。

ライブラリーを使って機材を選択し、DMX アドレスを割り当てることが出来ます。マウスを 1 回クリックするだけで 512DMX チャンネルまでパッチが可能です。全部で 8 つの DMX ユニバースが使用でき、パッチを一時的に解除するための「フライトケース」機能が使用できます。

Live fx（ライブ fx）
Programmer のエフェクトジェネレーターは三角関数をベースにしたもので、この関数原理を応用することによりサーキュレーション、プラットフォーム、レクトアングル機能などハイパフォーマンスエフェクトを可能にします。
さらに、オフセット／レシオ／ディレイなどの複合機能を使用することにより同一ラインやグループ上のインテリジェントライトに対してウェーブなどの追加エフェクトを簡単に付加することが可能です。

また、*ライブ fx* の設定はプリセットとして保存することができます。

その他の仕様
*カラーピッカー*をお使いいただくと、色や複数灯体に対する色のシーケンス(C/M/Y)を非常に簡単に作成することが可能です。*また、プリセット*で、色やポジションなどの設定を保存することが出来ます。

DMX 出力モニターにはカラーピッカーで作成された色が変換された DMX データが瞬時に反映されます。

Output（出力）

Programmer で作成されたデータは各種 DMX デバイス (Nano,Twilight)を通じて出力することが可能です。出力機材は同時に 8

個まで接続が可能で、合計 4096DMX チャンネルまで同時にコントロールすることが出来ます。また、種類の異なる機材を同時に接続することも出来ます（シリアル、USB、ISA、イーサネット等）。

Faderunit（フェーダーユニット）

e:cue のフェーダーユニットをお使いいただくと、現場でマウスを使ったオペレーションのストレスからデザイナーを解放し、より直感的な操作を行うことが可能になります。メタル製ジョグダイヤル×5、高機能フェーダー×13（サブフェーダー×12＋マスターフェーダー）各種ファンクションボタンを装備しています。輸送の際にも場所を取らないコンパクト設計です。

Media Engine（メディアエンジン）

シンプル DMX 出力に代わり、主に固定設備用に開発されたのがこの『メディアエンジン』、ユニバーサルメディアコントローラーです。*Programmer ソフトウェア*との接続にはイーサネットを使います。オプションのワイヤレスソリューションキットを導入することで、低価格でワイヤレス LAN としてお使いいただくことが出来ます。

Visualization（視覚化）

CAST 社のソフトウェア *wysiwyg* をお使いの皆様には、*Programmer* で作成した DMX データを直接 *wysiwyg* と同期することが可能です。外部ハードウェアのインターフェイスは必要がないため、wysiwyg が同じ PC であっても TCP/IP で接続されているリモート PC でも同一の環境となります。このような 3D レンダリングソフトウェアを使用することにより現場でのデザインワークを極力軽減し、より緻密なプランニングに役立てることができるでしょう

2.　安全にお使いいただくために

- 輸送の際には、製品の入っていた外装パッケージをお使い下さい。（衝撃から製品を守ります）
- 寒いところからオペレーションルームに製品を持ちこんだ時など、急激な温度の変化で製品が結露する場合があります。操作を始める前に、製品が完全に乾いた状態にあることをご確認下さい（室内の空気になれるまで、電源を入れないで下さい）。
- 塵や埃の多い場所や直射日光の当たるところで本製品を使用しないで下さい。
- テクニカルデータに従って、オペレーションルームの環境温度を守って下さい。
- 製品後方部に記述されている電圧と、お使いの地域の電圧に相違のないことを必ずご確認下さい。(Twilight は USB から直接電源を供給しているため、この限りではありません)
- 各種ディバイスをお使いになる時は、当社の指定する純正 AC アダプターもしくは電源コードをお使いください。
- ワイヤーにつまづいたり踏んだりしないようお気をつけ下さい。
- 落雷などの強電被害から機材やデータを守るため、そのような状況が予測される場合は全ての信号線を外して下さい。
- 機材の中に異物が挿入しないように注意してください。
- 外箱や電源コードの破損、液体や異物の挿入など緊急の際には機材のスイッチを切り、電源プラグを外してカスタマーサービスにご相談下さい。
- 本体の故障の際はご自身での修理を試みず、輸入代理店もしくはその指定するサービスプロバイダでお願いします。ユーザー自身で補修できるパーツ類は一切ありません。

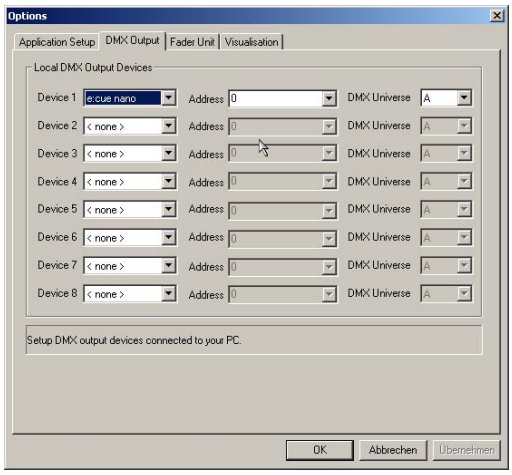
3. ショートガイド／クイックスタート

3.1. DMX 出力デバイスの初期設定

この機材をご利用になるには、まず DMX 出力デバイスを PC に接続しなくてはなりません。現在ご利用になれる製品は、nano（ナノ）、twilight（トワイライト）、media engine（メディアエンジン）の３種類です。

Nano をお使いになられる際は付属のケーブルを使ってお手持ちの PC の COM ポートに接続して下さい。また、電源は付属の AC アダプターからの電力供給となります。*Twilight* は USB に接続して使用します（USB パワーサプライとなっています）。
Twilight については、*ProgrammerCD* の中に入っているインデックスドライバーを使って各種 WindowsOS 対応のドライバーをインストールする必要があります。

次に、メニュー*EXTRAS/OPTIONS* を出して、*下図のようにデバイスを定義します。*プルダウンメニューでデバイスを決め、Adress でそのデバイスが使用するスタートチャンネルを定義します。



e:node（イーノード）と media engine（メディアエンジン） はイーサネットケーブルを使ってネットワークに接続して使います。そのため、まずシステムの管理者の指示をあおいで下さい。

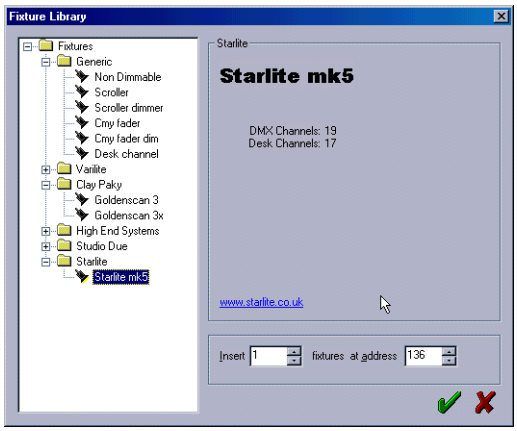
クロスケーブルで PC と *Mediaengine* を接続して下さい。TCP/IP プロトコルをまずインストールし、お手持ちの PC に有効化させることを忘れないで下さい。その後 *Mediaengine* に IP アドレスを割り当て、*メディアエンジンのディスプレイメニューSystem/Net* で調整して下さい。
(例)
Mediaengine 側 IP=192.168.123.1
PC(Programmer)側 IP=192.168.123.12
この時、上位 3 セクション(ここでは 192.168.123 まで)は、どちらも同一の値を取る必要があります。この値が違うとお互いのネットワークを認識しないことになりますのでご注意ください。
またサブネットマスクは完全一致することが必要です。
(例)255.255.255.0
E:node の IP アドレスは、ソフトウェアの *EXTRAS/SHOW PROPERTIES* を使って設定して下さい。

3.2. e:cue フェーダーユニットの初期設定

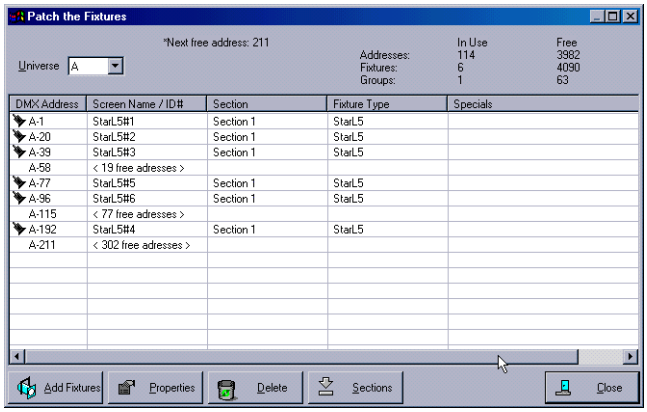
e:cue の *faderunit*（フェーダーユニット）は PC の COM ポートに接続します。Faderunit 本体後方部にある選択ボタンが[*SERIAL*]に設定されているか必ずご確認下さい。 ソフトウェアとシンクロ（同期）している時にはパワーLED が点滅します。
メニュー*EXTRAS/OPTIONS/FADERUNIT* を使って COM ポートを定義します。フェーダーユニットのパワーLED がコンスタントに点滅しているか確認して下さい。これでフェーダーユニットをお使いいただく準備が整いました。

3.3. 使用する DMX 機器の選択とパッチング

メニュー*FILE/PATCH* を選択すると、DMX 機器を選択できるパッチが面が出現します。このチャートに、現在選択されている灯体が表示されます。(デフォルトでは何もありません)
[*ADD FIXTURES*]ボタンを押すと、新たな DMX 機器を選択することが可能です。



リストからお使いになられる機材と台数を選択して下さい。
Programmer が自動的に DMX スタートアドレスを割り当てますが、これは後ほど変更することも出来ます。



プロパティを変更するにはまずディバイスを選択し、*PROPERTIES* を使ってプロパティのダイアログを有効にして下さい。
CLOSE でプロパティの変更が有効になります。

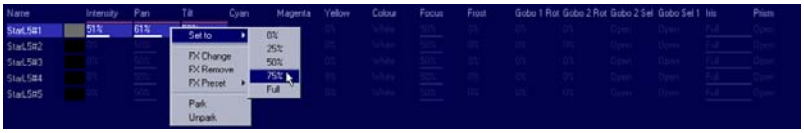
3.4. シーンを作成する

このソフトウェアでは、3 種類のビューを用いることが出来ます：

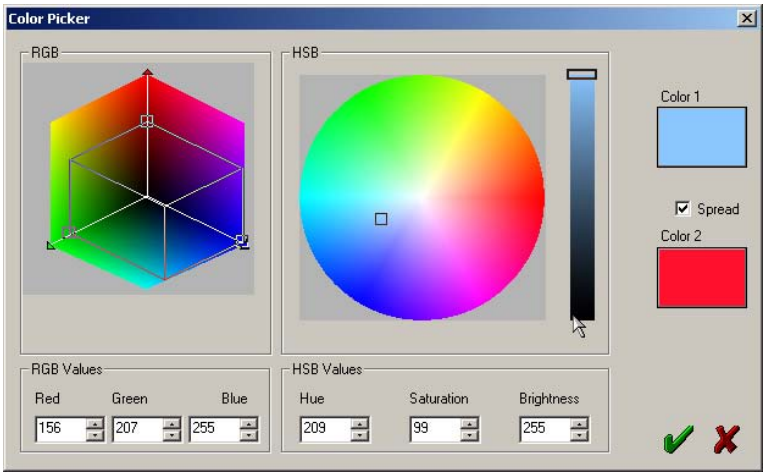
1. *programmer view（Programmer ビュー）*
2. *cuelist view（キューリストビュー）*
3. *stage output（ステージアウトプット）*.

Programmer ビューはシーンを設定するために使いますが、他のどのビューや DMX 出力よりも先に優先される優先権を持っています。[*VIEW/PROGRAMMER*] で Programmer ビューに変換することが出来ます。ナビゲーターウィンドウの左側に、ユーザーが先ほど選択した機材群が機材別にリストとして表示されます。マウスをクリックすると機材のリストが開き、任意のディバイスが *Programmer ビュー*（右側）にロードされます。機材のパラメーターはマウスを使って設定することが出来ます。

選択した機材のパラメーターだけが設定されます（ブルーマーク）。
もっと複雑な機能とプリセットは、右のマウスボタンを使ってパラメーターを有効にしなくてはなりません（コンテクストメニュー）



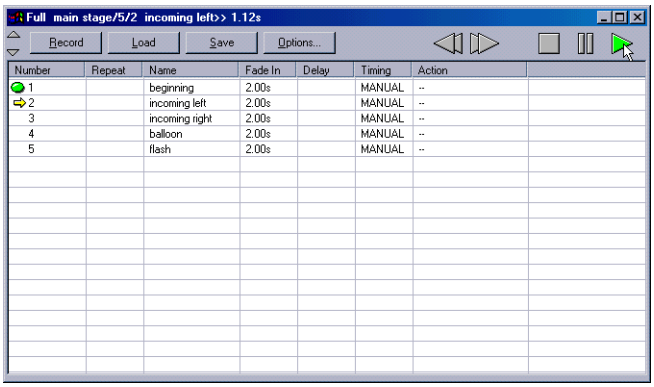
機材の名前の右側にあるフィールドはプレビューウィンドウです。機材の照度や色をシミュレート表示します。コンテクストメニューを使うとカラーピッカーを使用することも出来ます。



値の上方部にある赤いバーはフェーダーユニットのジョグダイヤルと直結しています。このジョグダイヤルを使って変化を施すことも可能です。

3.5. シーンの保存

シーンを保存するとナビゲーターウィンドウ左側のキューリストページ（*PAGE 1*）が有効になります。すべてのシーンが保存され、これをキューリストと呼びます。ひとつのキューリストには 4096 シーン登録することができ、また合計 240 キューリストを作成／操作することが出来ます。
選択されているキューリストは青い色にハイライト化されます。
CUE/RECORD でシーンは保存されキューリストチャートに表示されます。



指示時間、フェードイン／フェードアウトなどをシーンの設定で変更させることが出来ます（選択したシーンのコンテクストメニュー）。
Programmer ビューを使うと、他の機材を読み込んで新しいシーンを作成することも可能です。
注意： 使用した後は、必ず Programmer ビューの設定を解除して下さい（X 記号を使います）。

3.6. 保存したシーンを呼び出す

キューリストは *PLAY-BUTTON* で再生出来ます。シーンは DMX アウトプットディバイスに出力されます。*フェーダーユニット*を使うと、最初の 12 キューリストをフェーダーユニットのフェーダーを使って操作することが可能になります。

3.7. ウィザード

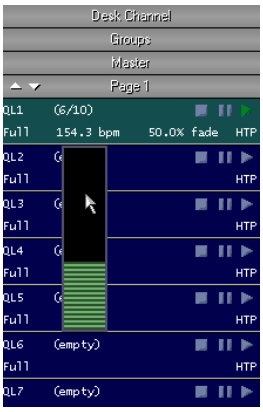
複雑なシーケンスは、ウィザードをお使いいただくと手早く作成することが出来ます。3 種類のウィザードを用意しています。

3.7.1. チェイサーウィザード

投射ライトはライトを集合化させると自動的に作成することが出来ます。例えば、すべてのライトを Programmer に読み込みます。すべてのディバイスの値を 0%に設定し、最初のライトを 100%にします。

これでチェイサーウィザードをスタートさせることが出来ます *(EXTRAS/CHASER WIZARD)*。ステップの数を入力すると（例えばライトの数）アシスタントは現在選択されているキューリストに自動的に必要なキューを割り当てます。リストをリクエストすると、自動的にスタートします。

チェイサーのキューリストは自動的にスピードコントロールを調整するオプションがつきます。キューリストの DPM 値のクリックして変更して下さい。PLAY キー(F12)を使ってミュージックビート（BPM ティーチイン）に変調することも出来ます。



3.7.2. スクロールウィザード

スクロールウィザードはスクロールテキストを作成するために使います。詳しくは以下の章をご参照下さい。

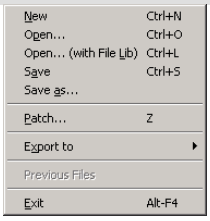
3.7.3. チェックイットウィザード

チェックイットウィザードは任意のタイプのライトのルーティーンをテストします。ムービングライトを選択して、チェックイットウィザードを立ち上げて下さい。キューリストはルーティーンで満たされています。これでリストをナノに送達することが可能になり、スタンドアローンモードでディバイスをテストすることが出来ます。

4. Programmer のメニュー構成

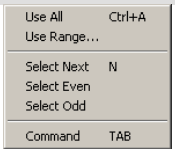
Programmer に存在するさまざまな機能ををより深くご理解いただくため、メニューについてご説明致します。

Menu File（ファイルメニュー）



New		新規ショー設定
Open		既存のショーを開く
Open (Lib)		既存のショーをライブラリーで開く
Save		現在のショーを保存
Save as		現在のショーを名前を変更して保存
Patch		DMX 機器ををパッチするためのダイアログ（第 5 章をご参照下さい）
Export to		キューリスト、トリガー設定、ショープランのパッチをテキストファイルにエキスポート
Exit		アプリケーションの終了

Menu Edit（編集メニュー）

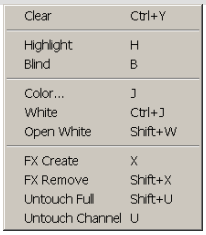


Use All	すべてのデバイスを選択
Use Range	デバイスを選択するためのダイアログを開く
Select Next	次のデバイスを選択
Select Even	偶数番号のデバイスを選択
Select Odd	奇数番号のデバイスを選択
Command	コマンドラインの解説

Menu Browser（ブラウズメニュー）

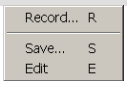
Fixtures	次のディバイスをナビゲーターに表示する（7.1 をご参照下さい）
Groups	グループを表示
New Group	選んだ灯体群を新しいグループとして設定する
Masters	スピードマスターメーターを表示
Cuelists	現在編集中のキューリストを表示

Menu Programmer（Programmer メニュー）



Clear		Programmer モードでの作業を終了する
Highlight		選択したライトの照度を一時的にフルに設定
Blind		*Programmer* ビューの出力を削除する。*Programmer* は有効のままですが、接続したディバイスへの影響は失われます（7.5 をご参照下さい）
Color		選択したディバイスの色設定のためのカラーピッカーを表示させる（7.8 をご参照下さい）
White		選択したディバイスの色設定を白にする
Open White		すべての灯体のチャンネルに影響を及ぼし、デフォルト値に設定します。フォーカスチャンネル（例えば、パンやチルト）には何ら影響はありません
FX Create		エフェクトジェネレーターを呼び出す（選択したパラメーターのエフェクト）
FX Remove		選択したパラメーターに対するエフェクトを解除する
Untouch Full		選択したディバイスのすべてのチャンネルパラメーターに変化を施さないように設定する
Untouch Channel		選択したディバイスのひとつのパラメーターに変化を施さないように設定する

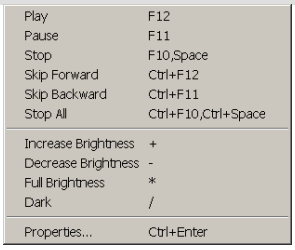
Menu Cue（キューメニュー）



Record	現在の Programmer の設定を、キューとしてキューリストに保存する（8.1 と 7.10 を参照して下さい）
Save	Programmer の内容をキューリストに保存する（8.5 を参照して下さい）

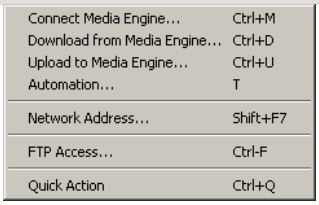
Edit	選択したキューの内容を Programmer へロードする（編集のため）

Menu Cue List（キューリストメニュー）



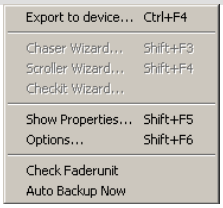
Play	▶	キューリストを再生（8 をご参照下さい）
Pause	⏸	キューの一時停止、もしくは前に戻る
Stop	■	キューリストのプレイバックを停止する
Skip Forward		次のキューでスタンバイさせる
Skip Backward		前のキューでスタンバイさせる
Stop All		すべてのキューリストの再生を停止
Increase B.		キューリストのサブマスター設定を増加（8 をご参照下さい）
Decrease B.		キューリストのサブマスター設定を縮小
Full B.		キューリストのサブマスターを 100%に設定する
Dark		キューリストのサブマスターを 0%に設定する
Properties		現在のキューリストのプロパティを表示する（8.7 をご参照下さい）

Menu Engine（Mediaenigine メニュー）



Connect ME		TCP/IP を使って e:cue メディアエンジンと同期する
Download		メディアエンジンのショーを Programmer にダウンロード
Upload		Programmer のショーをメディアエンジンにアップロード
Automation		Mediaengine 上で行われるオートメーションプログラムの設定を行う
Network adress		Programmer 側の IP アドレスを指定する
FTP Access		Mediaengine メディアデータ（JPEG、MP3）を FTP 経由でアップ／ダウンロードする
Quick Action		Mediaengien にコマンドで対応する

Menu Extras（エキストラメニュー）



Export to		接続している DMX アウトプットディバイスにショーをエキスポートする(Nano,E:com)
Chaser Wizard		チェイスプログラムのウィザードを呼び出す
Scroller Wizard		マトリックスプログラムのウィザードを呼び出す
Checkit Wizard		テストシーケンスウィザードを呼び出す（自動ムービングライトテスト）
Show Properties		現在のショーの設定を表示する

Options	各種オプションの設定を行う
Check Faderunit	フェーダーユニットの各種ボタン／フェーダー機能をチェックする
Auto Backup now	自動バックアップの設定を行う

Menu View（ビューメニュー）

Media Directory		メディアエンジン／メディアディレクトリの内容を表示する
Preset List		現在のプリセットリストを開く
DMX		現在の DMX 出力を数値で一覧表示する（7.2.3 をご参照下さい）
CD Player		タイムコードリセプションの CD プレイヤーを再生する（8.8 をご参照下さい）
Cuelist		選択したキューリストのビューウィンドウを表示する（8 をご参照下さい）
Fader		サブマスターとグランドマスターのビューウィンドウを表示する（8 をご参照下さい）
Jog Unit		ジョグダイヤルウィンドウを表示する
Toolbar		ツールバーを表示する
Statusbar		ステータスバーを表示する
Programmer		Programmer ビューに切り替える
Cuelist content		キューリストコンテントに切り替える
Output		DMX 出力画面（ビューモード）に切り替える

5. DMX 出力デバイスのアライメント

ライブなどのモバイルオペレーションでは、DMX 出力デバイスを使って行います。最大 8 個までの出力デバイスを接続できるほか、Mediaengine も同時接続することが可能です。また、Mediaengine をメディアコントロールセンターとして使い、個別にプログラムすることも可能です。

現在ご利用可能な端末：

▸ e:cue nano（ナノ）
Nano は COM ポート（シリアルポート）に直結させる DMX 変換デバイスです。

▸ e:cue twilight（トワイライト）
Twilight は USB ポートに直結させる DMX 変換デバイスです。

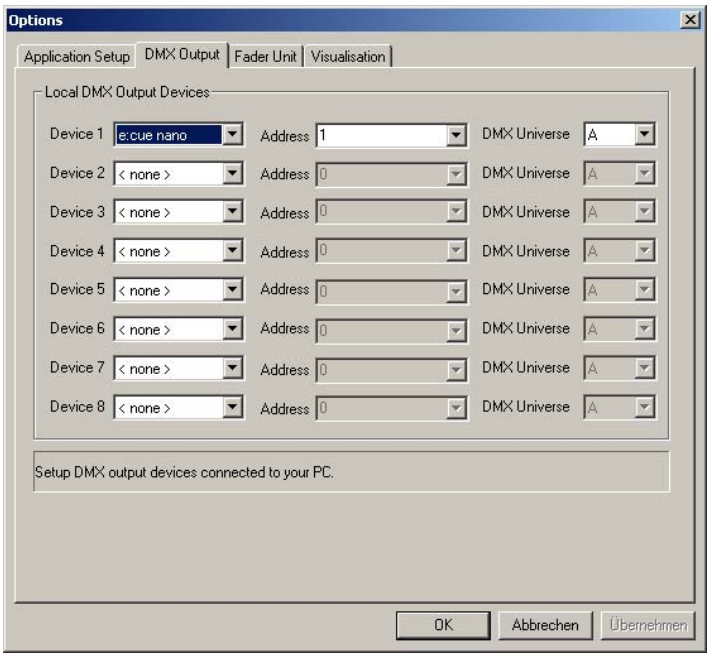
▸ e:node（イーノード）
e;node はイーサネット環境で使用する DMX ゲートウェイで、10 個のファンクションキーから構成されています。このファンクションキーに各種キューリストやオプションコマンドを割り当てて使用するデバイスです。Programmer の DMX データはネットワークの TCP/IP 経由で e:node へ伝送されます。

▸ e:cue media engine（メディアエンジン）
*メディアエンジン*は多彩なインターフェイスのメディアコントロール。e:cue の Programmer を使って完璧なプログラムが可能です。

5.1. 出力デバイスを定義する

DMX 出力デバイスを定義するには、EXTRAS/OPTIONS/DMX OUTPUT のダイアログを開いて下さい。 ソフトウェア上の DMX ユニバースは各出力ディバイスに自由に割り当てることが出来ますが、ダイアログでは現在 PC に接続されている機材しか定義することは出来ません。出力デバイスのうち、e:node はネットワークデバイスとして統合され、ショーのパーツとして取り扱われます。アドレスは EXTRAS/SHOW PROPERTIES/DMX NODES のダイアログで定義することが出来ます。

プログラムされたショーは後ほど mediaengine や Nano などに伝送することが出来ます。また DMX 出力を e:node に送ることも出来ます。



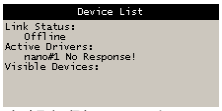
▸Device n(ディバイス 1～8)
DMX 出力ディバイスを定義する。
▸Adress(アドレス)
接続しているポート（COM1 から 8）もしくは USB ディバイス（1 から 8）を選択する。
▸DMX Universe(ユニバース)
どの DMX ユニバースを使用するのか定義する。

☝	アドバイス：複数のディバイスを同じ DMX ユニバースで同時に使用することが出来ます。

☝	アドバイス：複数のディバイスを同じバス（USB、シリアル、イーサネット）に繋げることが出来ます。

もし *e:cue フェーダーユニット* が PC に接続してあれば、レジスターカードフェーダーユニットを使って COM ポートを定義することが出来ます。

5.2. モニターステータス



上部左側のモニターステータスを使うと、現在接続しているデバイスの概要が一覧できます。また、ネットワーク環境も同時表示するため mediaengine や e:node との同期状況なども一目でわかります。

5.3. キューリストを出力ディバイスにエクスポート

Programmer を使用して作成したショーをデバイスに伝送することが出来ます。

2 種類のエキスポートマネージャーの使用が可能です。

ポータブル出力デバイス(nano,e:node など)へのエクスポートダイアログを表示します。

まず機材が定義されているか確認して下さい（5.1 を参照して下さい）。記憶型ポータブル出力デバイスの場合、最大 250 キューでかつ 1 キューリストのみがエクスポートの対象となります。したがって、現在のショーで設定している全てのキューリストをアップロードできるわけではないので注意が必要です。

	アドバイス： まず確認する キューをすべてディバイスに伝送できるかどうか、まず確認して下さい。

TCP/IP を使って *Mediaengine* と Programmer をオンライン状態にします。オンライン状態になると、画面左側のステータスモニターが変化し、オンラインになったことが表示されます。

そのためには、まず IP アドレスを*メディアエンジン*に認識させなくてはなりません。ショーを*メディアエンジン*に転送するにはメニュー ENGINE/UPLOAD を使用します。

5.4. Nano（ナノ）

Programmer のデータを DMX データに変換する一番簡単な方法は、*nano* をお使いいただくと非常に便利です。*nano* はとても小さい、防水加工の箱型の形状をしており、ラップトップから直接 256DMX チャン

ネルの使用が可能となります。RS232 を使って PC やラップトップに接続し、同時に複数台のナノをご使用いただくことも可能です。

Nano の特筆すべき点としては、*スタンドアローンモード*でお使いいただけるということです。つまり、nano の記憶領域にキューリストを入れ、PC から外し、nano の電源を入れると自動的にキューリストの再生を始めることが可能になります。ショーウィンドウなどでシングルプログラムをコンソール無しで再生するといったアプリケーションになどに最適といえます。

Nano をスタンドアローンモードでお使いいただくには、以下のことにお気をつけ下さい：

- 最大 32DMX チャンネル
- 1 キューリスト
- ライブ fx(エフェクト)機能は使えません.。
- フェードのソフトスタート／ソフトストップは利用出来ません。

☝	アドバイス：ショーを伝送する前に、 まず確認する を基本にデータの確認をして下さい（5.3 をご参照下さい）

正面の赤い LED は現在のディバイスの状態を示しています。PC 上の Programmer に接続されている時、LED は点灯しています。キューがディバイスのメモリーに読み込まれている時、LED は点滅しています。

データが伝送されるとLEDは緑色になります。

テクニカルデータ → 追記をご参照下さい

5.5. Twilight（トワイライト）

e:cue Twilight はUSBポートを使用するDMX出力デバイスです。

*nano*との相違点はDMXユニバースのすべての512DMXチャンネルの使用が可能であることです。しかしスタンドアローンでの使用は出来ません。*Twilight*はnanoと同じくアルミニウムの箱型の形状をしていますが、PCへの接続はUSBを使います。ソフトウェアを使って一度に最大8個の*twilight*を操作することができ、この場合、Programmerで利用可能な4096DMXチャンネル全部の使用が可能となります（ユニバースAからH）。

*トワイライト*の電源はUSBバスから供給されます。3つのLEDが機材の状態を表します。

POWER(電源ステータス)
電源がきちんと供給されているかどうか表示します。

USB
ディバイスのUSBドライバーがオペレーションシステムの指示に従って動作しているかどうか表示します。PCがアプリケーション起動中twilightを認識していれば点灯します。

DMX
DMX伝送データに何らかの変化が生じた時に点滅します。
非常に早い間隔で点滅し、実際にDMXデータが送出されているかどうかの目安になります。

テクニカルデータに関しては追記をご参照下さい。

5.6. E:node（イーノード）

E:nodeはイーサネットをDMXゲートウェイとして機能します。最大1024DMXチャンネルまで再生可能で、*Programmer*だけでなく*mediaengine*のデータも伝送することが出来ます。

ライブ
ライブオペレーションの際には、DMXデータ受信のためにTCP/IPネットワークのIPアドレスがきちんと設定されていなくてはなりません。このアドレスは、ProgrammerSHOWPROPERATIES/DMX NODESを使って登録します。

ESC キーと SEL を 3 秒間押し続けるとメニューモードになります。
UP/DOWN キーを使って IP のコンフィギュレーションを選択し、
ENTER で確定して下さい。UP/DOWN キーを使うと IP アドレスも簡単に設定することが出来ます。

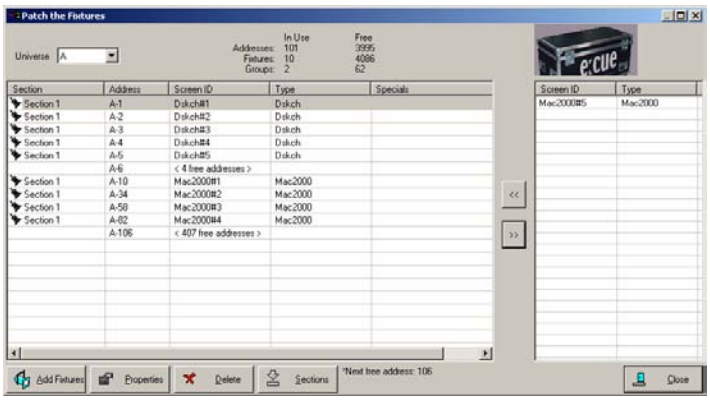
スタンドアローン
Programmer を使ってキューリストを e:node のメモリー領域に伝送することが出来ます。パネル上のファンクションキーを使って操作できますが、この機能はまだ開発中です。(2003 年 1 月現在)

☞	アドバイス：多様なディバイスが今現在も開発中です。 詳しくは e:cue のウェブサイトでご確認下さい。 www.soundfarm.co.jp/ecue(日本語オフィシャルサイト) www.ecue.tv(英語オフィシャルサイト)

6. DMX ターミナルの定義

Programmer を使ってショーを作成するには、まず出力デバイスにどのような灯体を接続するのかを定義しなくてはなりません。DMX ターミナルの大部分のライブラリーは自由に設定することが出来ます。接続する全ての DMX 機器には固有の DMX アドレスを割り当てなくてはいけませんが、この作業を『*パッチする*』と言います。

*パッチボタン*を使うと 以下のダイアログが表示されます。



このチャートには、使用可能な DMX アドレスを検出して、定義されたすべての DMX 灯体のリストが表示されます。

▸Universe（ユニバース）
DMX 灯体ににどの DMX ユニバースを使用するか定義します。
Programmer では、全部で A から H の 8 ユニバースの使用が可能です。

オプションに関しては、queue2-5 をご参照下さい。

▸DMX Address（DMX アドレス）
DMX 灯体がパッチされる DMX ユニバースと DMX アドレスを表示します（例：ユニバース A、アドレス 1）。

DMX アドレスがユニバースに設定されていない時には、未使用領域のキューで表示されます。

▸Screen Name / ID#（スクリーンネームとスクリーン ID 番号）
各灯体には*スクリーンネーム*と*スクリーン ID 番号*を割り当てることが出来ます。この名前は後ほどスクリーンに灯体の名前として表示されます。スクリーン ID 番号は灯体をソートする時に使います。

全ての DMX 灯体は DMX アドレスではなく、*スクリーン ID 番号*を使ってソートされるので、スクリーンビューは灯体の場所から DMX アドレスへ移行することが出来ます。（6.2 をご参照下さい）

▸Section（セクション）
複数以上の灯体を操作する時には、セクション別に小分けすることが出来ます。任意のセクションはスクリーンビューで自動的に展開したり格納することが出来ます。また、隠すことが出来ます。8 セクションに任意の名前を付けることが出来ます（6.3 をご参照下さい）。

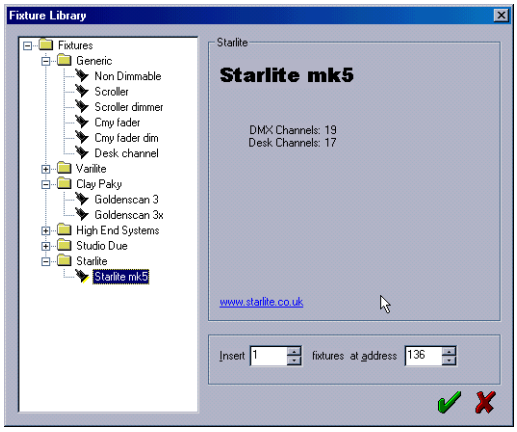
▸Fixture Type（灯体の種類）
オリジナルの灯体の種類が表示されます。*スクリーンネーム*の設定が施されている時などに便利です。

▸Specials（スペシャル）
もしプリヒートやインバートなどの設定がしてあれば、ここにリスト表示されます（6.2 をご参照下さい）

もしショーに使われている灯体群が一時的に必要なくなった場合、 >> や << ボタンを使って フライトケース へエクスポートすることが出来ます。必要な時には取り出すことが出来、新たにパッチする必要はありません。

6.1. DMX 機器を追加する

*Add Fixture（灯体を追加する）*ボタンを使うとライブラリーにディバイスを追加することが出来ます。

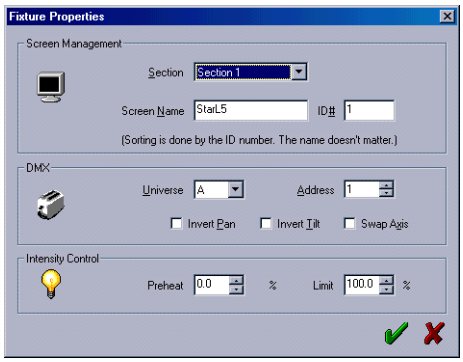


ディレクトリツリーをご覧いただくと使用可能な機材を一覧することが出来ます。ダブルクリックして利用可能なディバイスのリストを開いて下さい。ディバイスを選択すると機材の台数と DMX スタートアドレスを挿入し、設定することが出来ます。スタートアドレスは現在のリストから計算して割り当てられますが、手動で新しいアドレスを設定することも出来ます。ダイアログで選択したユニバースの設定をします。2 台めのディバイスの DMX アドレスは自動的に割り当てられますが、変更することも可能です。

6.2. DMX 灯体の特別設定（オプション）

Properties のボタンを使うと灯体オプションのダイアログが開きます。各灯体に特別な設定を施すことが出来ます。

	アドバイス：複数の灯体を選択し、一括して変更を施すことも出来ます。



▸Section（セクション）
灯体をセクションに割り当てます（6.3 をご参照下さい）。

▸Screen Name（スクリーンネーム）
灯体に名前をつけて下さい（例：backspots 等）。ディバイスに名前をつけておくと、プログラムする時に便利です。ID で、複数の灯体に連続した番号をつけます。ビューモードは、ID 番号でソートされます。

▸Universe（ユニバース）
DMX ユニバースを定義して下さい。Programmer で A から H、全部で 8DMX ユニバースの操作が出来ます。

▸Address（アドレス）

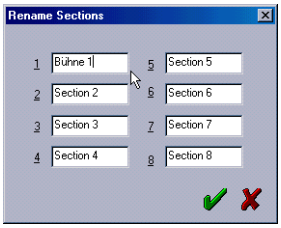
DMX スタートアドレスを定義して下さい。チャンネルはライブラリーで自動的に割り当てられます。

▸Invert Pan / Tilt / Swap Axis（パン、チルトの逆転／軸の交換）
ムービングライトのパンとチルトの自動逆転の設定をします。パンとチルトの置き換えもこの項目を使って設定します。

▸Preheat / Limit（プリヒート／リミット）
ディバイスの照度チャンネルを定義します。DMX 出力のプログラム設定の間は、最低値が最高値を超えることはありません。

6.3. セクションを使う

ステージや集合体の複数以上の灯体を使用する際には、セクションに小分けすると便利です。後で特定のセクションの灯体をグループとして選択することが出来ます。8 セクションに任意の名前を付けることが出来ます。またスクリーンビューをセクションに小分けすることも出来ます。



7. Programmer を使ってみよう

Programmer は以下の 3 種類のビューを表示させて使うことが出来ます：

Programmer ビュー、キューリストコンテント、ステージアウトプット

ビューを変更するには、メニューバーにあるボタンをお使いいただくか、ビュー内のロゴをマウスで右クリックして下さい。

1.1. ナビゲーター

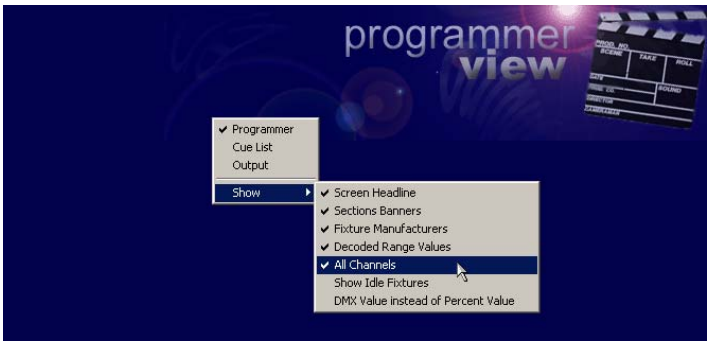
画面左側に位置するナビゲーターを使えば、すべての灯体や所属グループ、キューリストにアクセスすることが出来ます。

特定のレーンはヘッドラインとマウスで利用することが出来ます。この場合スクリーンのスペースは少なくなり、ビューモードのすべすはナビ

ゲーターのヘッドラインで有効になります。ナビゲーターのヘッドラインをマウスで右クリックして、 *Force Short View* を選択して下さい。英語バージョンではナビゲーターウィンドウのことを『ブラウザ』とも呼びます。

7.2. Programmer の 3 つのビューモード

Programmer のビューは変更することが出来ます。コンテクストメニューの Show を選択して下さい。



最初のオプションは、現在使われるデータを設定します。現在使われていないディバイスは Idle Fixtures となります。
パーセントではなく実際の DMX 値を表示させるには、ライブラリーデータ表示の最後の入力を有効にしなくてはいけません。

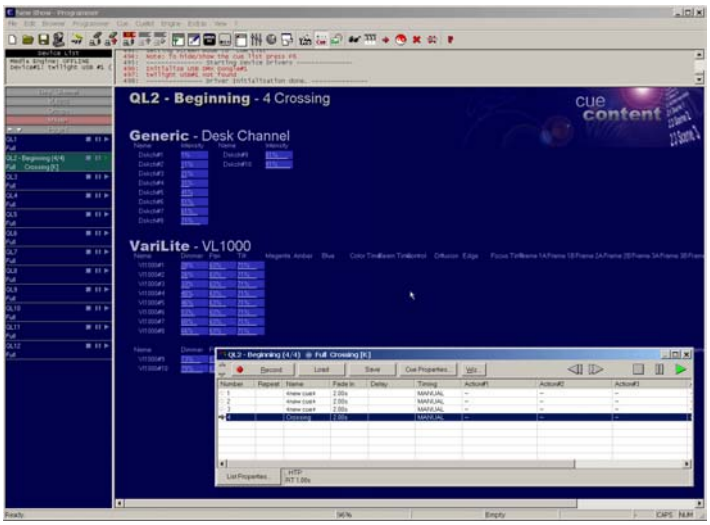
7.2.1. Programmer ビュー

このビューモードは、後でキューと呼ばれるシーンを作成するのに使います。ディバイスはすべてスクリーンに表示され、パラメーターで使用することが出来ます。

	注意：Programmer ビューにデータが含まれている時には、そのデータが DMX 出力に対して何よりも先に優先権を持っています。 従って、キューリストを再生中にプログラムモードで使用しているとプログラムモードが優先され、灯体の DMX 値は LTP チャンネルに関してはプロフラムモードのものが採用されることになります。 ボタン をお使いいただくと、Programmer の現在の設定は解除され、キューリストが再度 DMX 出力のアクセス権を得ることになります。

7.2.2. キューリストコンテント

ビューモードのキューリストコンテントには、保存されたキューのキューリストが表示されます。キューリストは、ナビゲーター左側の、カテゴリーPage1 に表示されます。キューリストをダブルクリックするとコンテントのウィンドウを開きます。選択したシングルキューの内容が、コンテントウィンドウに表示されます。



Play / Pause / Stop を使うと、キューリストのシーンが有効になります。

キューリストを編集するための詳しい説明は 8 をご参照下さい。

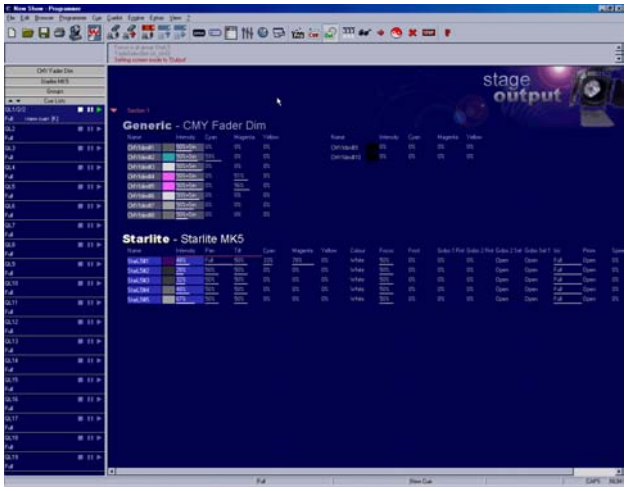
選択したキューの内容はメインウィンドウに表示されます。

7.2.3. ステージアウトプット

*ステージアウトプット*ビューは現在の DMX 出力を操作するために使います。

	注意：Programmer ビューにデータが含まれている場合、このデータが DMX 出力に対して優先されます。キューリストコ

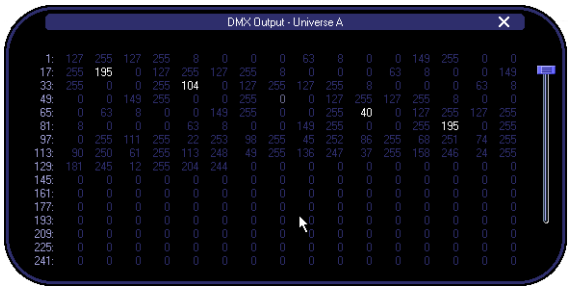
	ンテント同様、LTP チャンネルはプログラムモードに完全に支配されることになります。 このボタン を使うと Programmer ビューの現在の設定は解除され、キューリストが再度 DMX 出力のアクセス権を得ることになります。



	アドバイス：スクリーン上部にあるログブックは、右のマウスボタンでプレビューモードに変換されます。

DMX データを視覚的にブラウズするには、DMX モニターを使って下さい。メニューVIEW / DMX、もしくは F4 を使います。

ツールバーのこのボタン をお使いいただいてもモニターを表示させることが出来ます。



コンテクストメニューをお使いいただくと、マウスの右ボタンを使って DMX ユニバース（A から H）を選択することが出来ます。マトリックスエフェクトモードが有効な時には、変更を加えた DMX チャンネルが点滅します。他のチャンネルは黒っぽい色で表示されます。

7.3. 新しいシーンのために灯体を選択する

新しいシーンを設定するには、シーンに使うディバイスをまず *Programmer* ビューに読み込んでやらなくてはいけません。使用したい灯体をナビゲーターで選択し、機材を選んでクリックして下さい。

コンテクストメニューと右のマウスボタンで他のディバイスを Programmer に読み込むことも出来ます（Use all / Use range）。

灯体の選択を手早く行うために、数種類のディバイスをグループ分けすることも出来ます。グループ化し判別しておくと Programmer に読み込んだり、マウスを一度クリックするだけで有効化することが出来ます。

グループ化するためには、グループ分けしたい灯体をプログラムモード中で選択し、STRG+G で新しいグループが追加されます。

セクション分けされた各灯体はスクリーンに表示されます。セクションの割り当てはメニュー*Patch* で変更することが出来ます（6.3 をご参照下さい）。

ディバイスをソートする時には、*スクリーン ID 番号*を入力して下さい（Patch をご参照下さい）。

*e:cue フェーダーユニット*が接続されている時には、 ジョグダイアルの値はパラメーター上部の赤いバーに表示されます。フェーダーユニットのジョグダイアルを使うとパラメーターを直接変更することが出来ます。

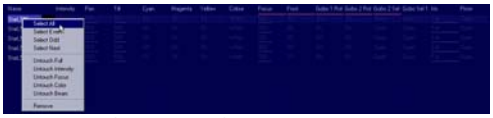
このボタン で*ジョグダイアル*のウィンドウが開きます。この場合、*ジョグダイアル*の割り当ては、クリアーテキストで表示されます。

まず各ディバイスの名前と ID 番号が表示されます。次に現在の色、明るさの出力（プレビュー）、パラメーター、カラー設定が表示されます。

クリック＆プル“（マウスでパラメーターをクリックし、そのまま左右に動かす）でパラメーターを直接変更することも出来ます。

	注意：シーンを保存する際には、マークのついている値だけに変更がなされます（touched チャンネル）。複数のキューを同じディバイスにアクセスさせることも出来ます（8.1 をご参照下さい）。

パラメーターをマウスで右クリックするとコンテクストメニューが表示されます。これを使うとゴボや照度などの設定を直接行うことが出来ます。



Untouch（アンタッチ）機能を使うと、選択した変更を手動でキャンセルすることが出来ます。

プレビューウィンドウで、出力される照度と色を確認することが出来ます。コンテクストメニューを使うとカラーピッカーを使って直接色を選択することが出来ます。

クリック＆プルでパラメーターに変更を施すことが出来ます。

複数のディバイスが選択されている時には、すべてのパラメーターが同じように変更となります。保存する際にどのチャンネルに変更が施されたか確認することが出来ます（*Touch / Untouch* 機能）。

	アドバイス：SHIFT キーを同時に押すと、もっと細かく設定することが出来ます。

	アドバイス：STRG キーを同時に押すと、もっと値を広げることが出来ます。

コンテクストメニューを右マウスクリックでもっと繊細な設定が可能になります：

Set to	リストに表示されているガイド値に従ってチャンネルを設定する
FX Create	エフェクトジェネレーター*ライブ fx* を呼び出す

	（7.9 をご参照下さい）
FX Remove	選択したチャンネルのエフェクトを無効にする
FX Preset	*プリセットしたエフェクトデータを操作する*
Park/Unpark	現在の値でチャンネルを凍結する チャンネルは P マーク（park）の後に赤色で表示されます
Untouch Channel Intensity	照度チャンネルに 変更を施さない が設定されます（キューとして保存される時にはこの限りではありません）
Untouch all ..	すべての照度チャンネルに変更を施さない
Create Intensity ..	選択した設定で新しいプリセットを造る

7.4. コマンドラインエントリー

TAB キーでコマンドラインが表示されます。このコマンドラインをお使いいただくと Programmer をマウスではなく直接コマンド操作することが出来ます。

入力例

1..5@20	DMX チャンネル 1 から 4 を 20%で設定
1+3+7@F	チャンネル 1、3、7 を 100%に設定
!<*Command*>	直接マクロにコマンドする

7.5. ハイライト機能

ハイライト機能 は、スクリーンの値で実際のライト灯体の場所をサポートします。ハイライトが有効な時、Programmer で有効になっているすべてのライトは点滅します。エフェクトとカラーチャンネルは、自動的に標準値に設定されます。

*ハイライト*モードが、Programmer のステータスに変更を及ぼすことはありません。ハイライトが有効になると、Programmer に設定した値が視覚化されます。

	アドバイス： ハイライトを有効にするにはまず H を押します。H を押してハイライトを有効にし、N（NEXT）を繰り返し押して実際のライトを探します。もう一度 H を押すとハイライトは無効になります。

7.6. ブラインド機能

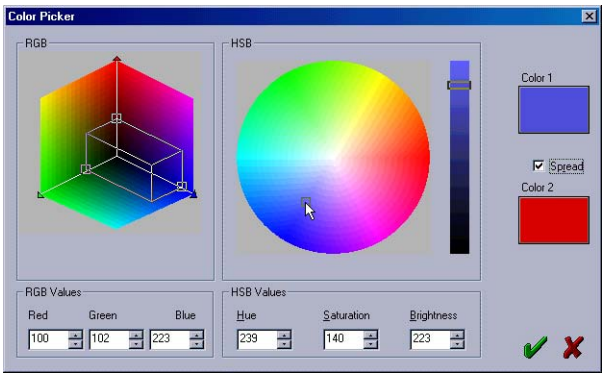
これまでのプログラムモードとキューリストモードでの DMX 優先権について簡単に述べるなら、プログラムモードが他のモードに被いかぶさっているということです。しかし、ブラインド機能を使うことによりこの原則を一時的に例外とすることが出来ます。ブラインドモードが有効な時には、Programmer のステータスが実際に接続されているライトに影響を及ぼすことはありません。ステージアウトプットビューの時にも同様です。

ブラインドモードを有効にするには、キーボードの B を押すか、眼鏡マークボタン をご使用下さい

7.7. フリーズ機能

フリーズ機能を使うと、ショーを一旦停止させ、その時のチャンネル値を DMX 出力へ送ります。Programmer の使用は出来ますが、ライブ fx で作成したエフェクトは凍結され、使用することはできません。

7.8. カラーピッカー



カラーピッカーを使うと簡単に色を選択することが出来ます。値を選択するだけでなく、 Spread 機能を使って色の再生をプログラムすることも出来ます。一種類以上のディバイスは、まず読み込んでおいて下さい。

プレビューウィンドウのコンテクストメニューからカラーピッカーを有効にするには J キーを使って下さい。

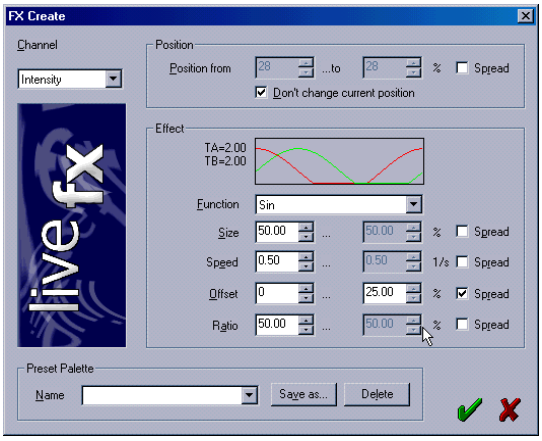
あ

☝	アドバイス：カラーピッカーをお使いいただいてもライトチャンネルの照度には何も変更を施しません。カラーピッカーをご使用いただいた後もライトが真っ暗な時には、照度チャンネルをご確認下さい。

☝	アドバイス：カラーピッカーは CMY（もしくは RGB）のカラー混合をサポートしているライトでのみお使いいただけます。（シアン、マゼンダ、イエロー、もしくはレッド、グリーン、ブルー対応）

7.9. ライブ fx エフェクトジェネレーター

ライブ fx は高性能なエフェクトジェネレーターです。各灯体の各チャンネルにエフェクトを施すことが出来ます。



数式機能を使って特定のパラメーターを変更することが可能です。

Size（サイズ）	パラメータの変化幅を設定する（%）
Speed（スピード）	パラメータの変化速度を設定する（Hz）
Offset（オフセット）	灯体グループ中のディレイ幅を設定する（%）
Ratio（比率）	パラメーター変化の開始点を決定します。

複数の同一種類灯体に対して*ライブ fx* を呼び出した際は、そのグループごとの値を設定します。

例：1 から 10 のムービングライトを Programmer に呼び出します。
チルトチャンネルをマウスで右クリックすると *FX 設定ダイアログ*が開きます。機能リストから一例として、*sin(サイン波)*を選択します。

*オフセットラインの Spread(スプレッド)*チェックボックスを ON にして、0 から 100.0 の値を入力して enter キーを押します。
ダイアログ中のグラフで点がモーションをかけるのと同時に、エフェクトがかかったムービングライトはウェーブモーションを描きます。

また、この場合開始点(Ratio)は 0 に設定されているので、常にパラメータ変化の開始点はもとの programmer から LiveFX を呼び出した時の値となります。この値を変更するには、programmer ビューの tilt の値を変化させます。

さらに、もしチルトのパラメータが 50%(DMX 値では 128)に設定されている場合、最大の振り幅は 50%です。このような場合に、Size(振り幅)を 50%に設定すると綺麗なサインカーブを描きますが、それ以上の値を指定するとサイン波の凸部分がグラフを逸脱し、逸脱している間は灯体はその最大値(0 もしくは 255-100%)をキープすることになります。

サウンド to ライト機能

この機能は、PC に入力された各種音源をソースにして、ある周波数のビートをシーケンスタイミングに変換する機能です。

サウンド to ライトのイコライザーバンド 1 から 16 を設定するにはダイアログをお使い下さい。例えば、現在のアナライザーの値は、1 から 16 のライト照度チャンネルから直接入力することが出来、イコライザーバンド X1 から X4 にプールされます。（バンド 1 から 4 →X1）

注意：パラメーターの比率と速度に相関関係はありません

サイズで伝達の強度を選択します

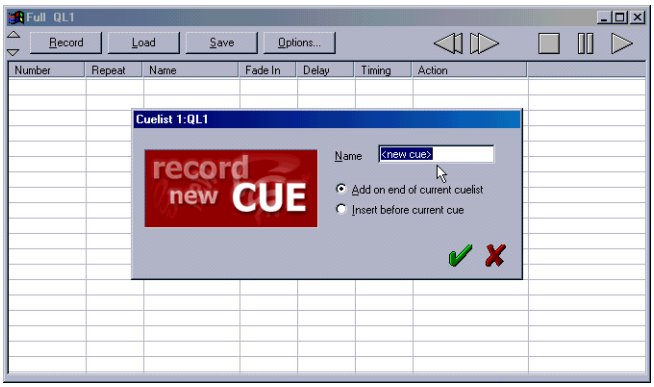
7.10. プリセット機能

プリセットを使うと、DMX 機材のプリセットを保存し、また簡単に呼び出すことが出来ます。

F3 でプリセット一覧を表示させたり隠したりすることも出来ます。

7.11. シーンを保存する（キューの登録）

Programmer ビューですべての設定が終わったら、R キー、もしくはメニュー*Cue / Record* でシーンをキューリストに保存して下さい。



各シーン（キュー）には任意の名前をつけることが出来ます。リストにシーンを追加する時には、リストの最後に追加するか、現在のシーンの前に挿入するかを選択することが出来ます。

シーンは常に現在のキューリストに追加されます。キューリストかナビゲーターの左側にある小さな矢印を使ってキューリストを選択して下さい。深緑色の印がつきます。

シーンからシーンへ移行する時にフェードさせるには、各キューで長さなどを設定することが出来ます（8.7 をご参照下さい）。

もしすでに保存されているキューに変更を施したい場合には、まずそのキューを Programmer ビューに呼び出して下さい（8.2 をご参照下さい）。シーンを選択して右クリックし、*Load* を選択します。

Programmer ビューで変更を施したら、キューリストウィンドウの *Save* ボタンで（S キー）保存します。

キューリストでは、チャンネルの設定のみが保存され、Programmer にマークされます（touched）。ここで Untouched を選択したチャンネルのパラメータはキューとして記録されません。応用例としては、別々の 2 つのキューリストを同時に走らせる場合を想定して、片方のキューにカラーやゴボなどのデータを、片方のキューにヘッド位置のみのデータをそれぞれ Untouch/Touch 機能を使って登録すると、レイヤーのように 2 つのキューがちょうど重なりあった状態で再生します。ヘッド位置などのキューデータをたくさん取るべき時などに便利な機能です。、

8. キューとキューリストを使用する

シーンはキューリストに保存されます。シーンとシーンの間のフェードの長さなどは、キューリストに個別に設定することが出来ます（8.7 をご参照下さい）

ナビゲーターがキューリストのコントロールファイルを表示します。

Play 、Stop、Pause ボタンを使ってキューリストを稼働させます。Play ボタンをもう一度押すとリストの次のキューを再生します（そのように設定されていなくてはなりません。8.7 をご参照下さい）。Pause を使うと演出の一時停止、またはひとつ前に戻ります。演出の終了を停止します。演出の間に使われるキューリストのアウトレットは、ステージアウトプットビューで確認することが出来ます。

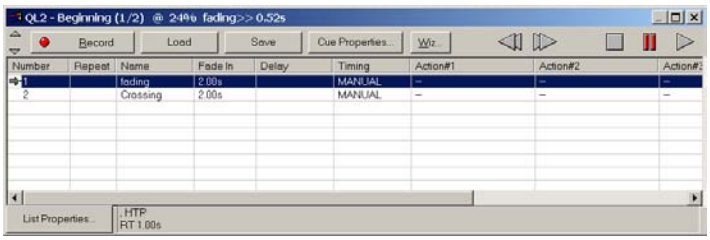
	アドバイス：同時に複数のキューリストをスタートすることが出来ます。

各キューリストには明るさを調節するサブマスターがついています。明るさの値をプレスすると有効になります。

キューリストの名前の下に、常に明るさは表示されます。

ナビゲーターを有効にすると、キー +、-、*、/ を使って、手動でのレギュレーションが可能になります。

このボタン を使うと、選択したキューリストのビューウィンドウが有効になります。コンテクストメニューで使用することも出来ます。



キューリストには現在のキューリストのすべてのシーンが表示されます。各項目の意味に関しては以下をご参考にして下さい。

▸Number（ナンバー）
キューの番号

▸Repeat（リピート）
演出の間に何度キューをリピートさせるか表示します。複数のキューをひとつのループとしてリピートさせることも出来ます。ループを組み合わせることも出来ます。

▸Name（名前）
保存の際にキューにつけた名前（8.1 をご参照下さい）

▸Fade In（フェードイン）
フェードインの時間を表示

▸Delay（ディレイ）
キューをスタートする時のタイムラグを表示

▸Timing（タイミング）
キューを読み込んだ後で、どの位待機するか表示します。マニュアルモードでは、Play ボタンが押されるまで待機した状態となります。

Wait（待機） モードでは、キューの再生時間は秒で定義されます。The measuring of time starts directly with the loading of the cue, independently from the chosen values for *Fade* and *Delay*.n. 再生時間が終了すると次のキューが読み込まれます。

*タイムコードモード*では、キューを読み込んだ後に、設定された値か、もしくはその値の大きいタイムコードのタイムが受信されるまで待機状態となります。タイムコードを受け取らないかぎり、キューが再生されることはありません。この場合、再度 Play を押すと、次のキューを再生します。

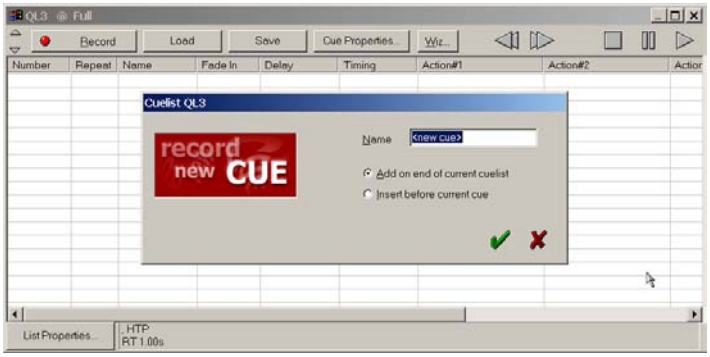
▸Action（アクション）

各キューを読み込んでアクションを繋げることが出来ます。これは通常、*メディアエンジンのスタンドアローンモード*で使用します（8.2、8.8 をご参照下さい）。

8.1. 新しいキューをつくる

シーンの設定が終わったらキューとしてキューリストに保存することが出来ます。R キーか、メニューCue/Record を使って下さい。

☝	アドバイス：R キーは次のキューを保存します！



各シーンには名前を付けることが出来ます。複数のシーンをリストに保存する時には、このシーンをリストの最後、もしくは最初に挿入することが出来ます。 ナビゲーターの中から希望のキューリストを選択してキューを挿入して下さい。

キューに変更を施すには、まず Programmer で読み込んで下さい（8.2 をご参照下さい）。キューの設定は Programmer ビューで可能になります。メニューCue / Save でキューリストに再度保存されます。

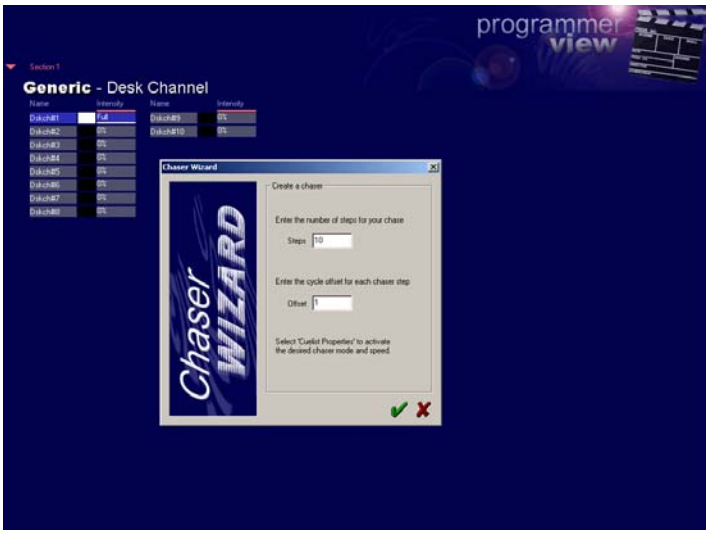
厳密に言えば、編集された変更だけが変更になります（タッチ機能）。通常は妨げになることなく、2 つのキューを同じ DMX 端末にアクセスさせることが出来ます。

8.2. チェイサーウィザード

チェイサーウィザードを使ってチェイサーをつくることが出来ます。

例：

Programmer ビューでライトを定義します。まず最初に照度チャンネルを 100%にセットし、他は 0%にして下さい。An activation of the *chaser wizards* with ten Steps so edits ten cues in the current cue list which set the lights on 100% brightness in an alternating way.



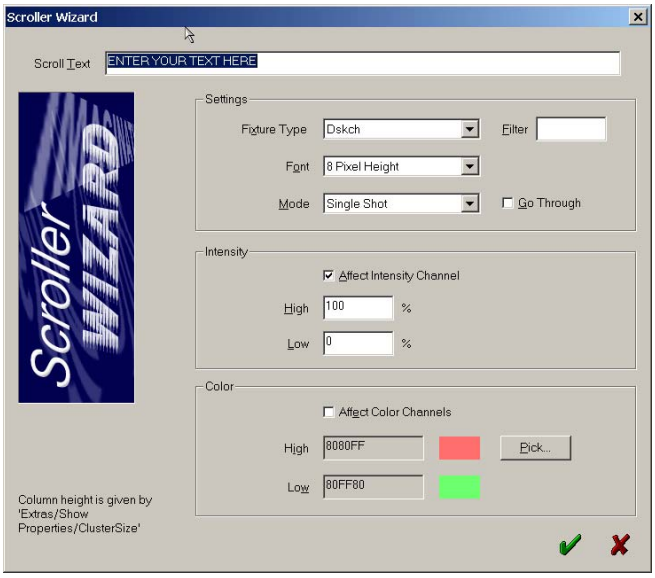
8.3.　スクローラーウィザード

スクローラーウィザードをお使いいただくと、簡単にスクロースステキストを設定することが出来ます。まず最初にライトを定義して下さい。ライトの傾向は ID 番号に左右されます。メニュー EXTRAS/SHOW PROPERTIES/CLUSTERSIZE で Y 方向へのライト数を設定することが出来ます。 The ID numbers in following are placed progressionally:

1	8	15	22	29
2	9	16	23	30
3	10	17	24	31
4	11	18	25	...
5	12	19	26	
6	13	20	27	
7	14	21	28	

(クラスターサイズを 7 で設定した時の例)

スクロールテキストを挿入するキューリストを選択して下さい。
SHIFT+F4 でスクロールウィザードを立ち上げます。



▸Text（テキスト）
スクロールさせる文章（言葉）を入力します

▸Fixture Type（灯体タイプ）
灯体の定義をします

▸Font（フォント）
フォントのサイズを定義します

▸Mode（モード）
スクロールさせる文章のエフェクトを定義します

Single Shot	文字が一括表示されます。
Coming from right	右側から文章が表示されます
Fly up	上部から文章が表示されます
Exit to Left	左側から文章が表示されます

Go through(ゴースルー)機能は、表示されたテキストを一旦とめることなく流してしまう機能です。例えば、 Coming from right を選択し、 Go through オプションを選択すると、文字は右から流れてきますがそのまま左に行き過ぎてしまいます。

▸ Affect Intensitiy Channel（ON/OFF の照度を変更する）

ここで定義した High スクロールテキストに使うチャンネルを定義して下さい。The array of lights can insist of dimmer channels and moving lights likewise. テキストはカラーチャンネルにのみ設定することが出来ます。

選択したシーンのウィザードのコンフィギュレーションを終了した後で、シーンのプリプログラムしたシーケンスを編集します。スクロールテキストのスピードは、キューのタイム設定の変更に影響されます。

構成のチェックはスクリーンステージアウトプットを使って行うことが出来ます。

☝	アドバイス：スピードマスターをキューリスト（キュープロパティ）に割り当てると、シーケンスのスピードも簡単に設定することが出来ます。

8.4. チェックイットウィザード

The checkit wizard generates test-routines for lights of arbitrary type. ムービングライトを選択してチェックイットウィザードを稼働させて下さい。通常キューリストはルーティーンと一致します。このリストをナノに伝達し、スタンドアローンモードでナノをチェックディバイスとしてお使いいただくことも出来ます。

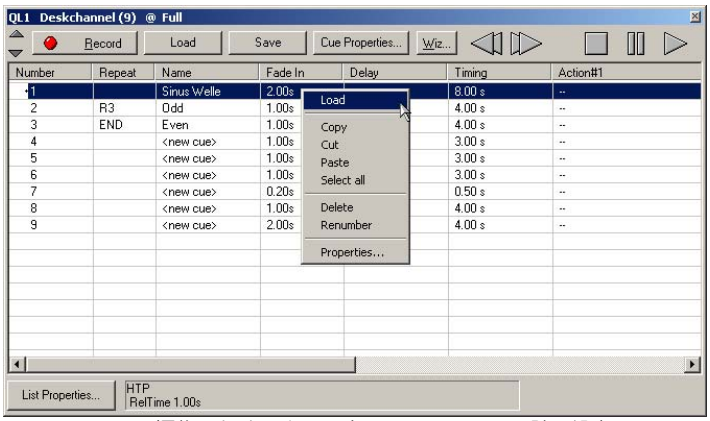
8.5. キューを変更する

もうすでに存在しているキューは Programmer ビューを使って編集することが出来ます。

キューリストウィンドウから ( ) キューを選択し、 Load ボタンを使って Programmer ビューへ読み込んで下さい。変更を施した後は Cue / Save を使って保存します。

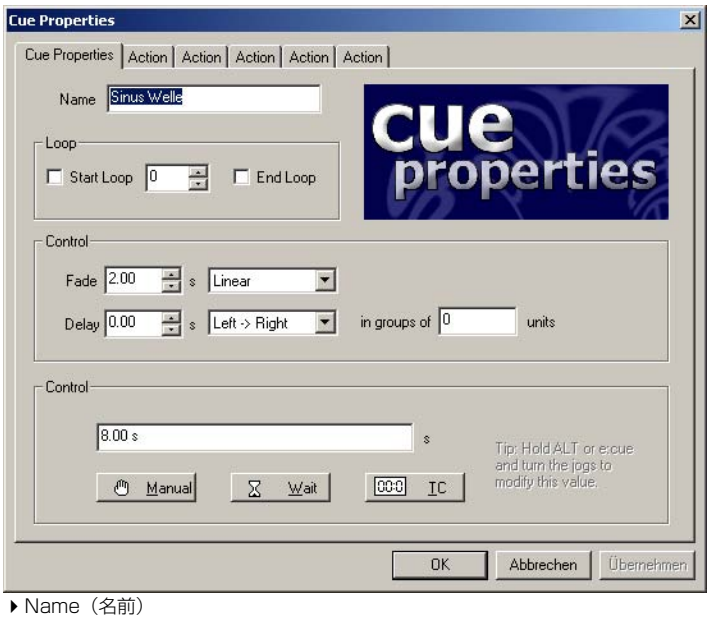
8.6. キューのプロパティ

キューリストビューウィンドウのキューコンテクストメニューを使うと（8 をご参照下さい）、保存したキューを編集することが出来ます：



Load	編集のためにキューを Programmer に読み込む（8.1 をご参照下さい）
Copy	クリップボードへキューをコピーする
Cut	キューをカットしてクリップボードに保存する
Paste	クリップボードからキューを挿入する
Select all	すべてのキューを選択する
Delete	キューを削除する
Renumber	新しいキューリストに番号を付ける
Properties	キューのプロパティ

*Properties（プロパティ）*を使うとプロパティの編集をすることが出来ます。複数のキューを選択して、同時に変更を施すことも出来ます。



▸Name（名前）
キューの名前を定義します。

▸Start Loop（ループを始める）
キューを再生すると選択したナンバーのループの繰り返しが始まります。ループを終了させるには、キューに *end loop* を設定します。ループは組み合わせて使用することが出来ます。

▸End Loop（ループを終了する）
ループの終わりを定義します。Start にチェックマークが入っているときには、end にもチェックマークを入れます。Start と End マークは常に同じ状態にしてやらなくてはなりません。Start と End が同じ状態で設定されていない場合には、ロジカルエラーとなります。

	アドバイス：複数のループを組み合わせて使うことが出来ます。

▸Fade（フェード）
シーンを挿入する時間を定義する

Linear	リニアフェード
Soft Motion	フェードをソフトに始め、ソフトに終わらせる（特にパンやチルトで使用します）
Soft Start	ソフトにフェードを始めます。
Soft Stop	ソフトにフェードを終わらせます

▸Delay（ディレイ）
ディレイはエフェクトとしても利用することができる便利な機能です。通常全ての灯体はキューの指示で一斉に動きますが、メニューを選択するとこの開始点を灯体によって遅らせることができるものです。

Left Right	最初のライトがすぐに変更され、最後のライトは設定された時間にディレイされます。
Right Left	最後のライトがすぐにフェードされ、最初のライトは設定された時間にディレイされます。
Middle Outside	中央のライトがすぐにフェードされ、他のライトは設定された時間にディレイされます。
Outside Middle	外側のライトがすぐにフェードされ、中央のライトは設定された時間にディレイされます。
Group units	Devides the delay properties into smaller groups. No value: all lights form one group.

▸Control（コントロール）
フェード終了後完全な位置になった時点からの灯体の次のアクションのための待ち時間を定義します。この待ち時間の後、次のメニューに移ります。

Wait（ウェイト）

この待ち時間の後、自動的に次のキューに移ります。

Manual（マニュアル）

この待ち時間の後、次のキュー呼び出しのアクションを待ちます。（キューリストビューダイアログの Play ボタン、もしくは e:cue フェーダーユニットの Play ボタンで次キューを実行します）。

Timecode（タイムコード）

*この待ち時間の後、受信タイムコードと比較して*読み込んだキューが設定された値か、もしくはその値より大きいタイムコードが受信されるまでは待機状態となります。タイムコードショーでは、tc 値に到達した時に次のキューの再生が始まります。

アドバイス：キューリスト左側下方にある learn timecode を有効にして下さい。すべてのキューを MANUAL 設定にし、タイムロケーションのタイムコードを受け取ったらキューリストの PLAY（再生）ボタンを押して下さい。これでタイムコードの時間が特定のキューに入力されます。

タイムコードを受け取らない限り、永遠に待機状態を続けます。次のキューへ移行するには、再度 Play ボタンを押して下さい。

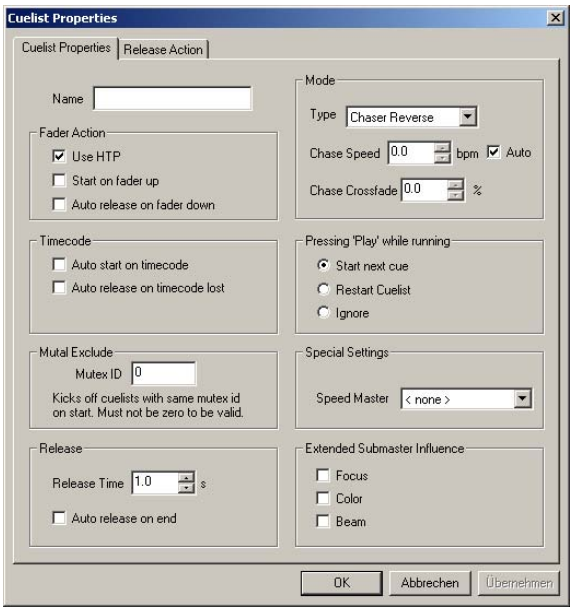
▸ Action（アクション）
e:cue のいわゆる「売り」である部分です。各キューを読み込んで、次のアクションを結合させます。（例：MP3 やビデオファイルの演出）。これは特に、*Mediaengine のスタンドアローンモード*でサポートされている機能です（8.2、8.8 をご参照下さい）Programmer を使うと、これらのアクションをライブオペレーションで使用することも出来ます。ウィンドウズメディアプレーヤーとの統合を有効にします。

（メディアプレイバックを操作するには、マルチメディアパックが有効でなくてはなりません。このマルチメディアパックは、フェーダーユニットが PC に接続されている時にのみ使用可能となります。）

e:cue メディアエンジン に接続されたディバイスとオーディオ／ビデオプレーヤーは、キューアクションによって操作されます（8.8 をご参照下さい）。

8.7. キューリストのプロパティ

ナビゲーター／キューリストのコンテクストメニューを使うと、キューリストプロパティの編集をすることが出来ます。



▸Name（名前）
キューリストの名前を定義します。

▸Use HTP（HTP を使用する）
2 つのキューリストが同じ灯体にアクセスする場合、どちらを使うか定義します。Http では大きい方の値が送られます。

例外：チャンネルがライブラリーエディターで LTP に定義されている時には、任意のキーが関連しているチャンネルに、最後の値が設定されます。

HTP: 一番高い値が優先されます
LTP: 一番最後の値が優先されます

▸Start on fader up（フェーダースタート機能）
フェーダーユニットの該当するサブフェーダーを 0 からあげるとキューリストが実行されます。

▸Auto release on fader down（フェーダーダウンのオートリリース）
サブフェーダーを 0 にするとそのフェーダーでプレイ中のキューリストが停止し、DMX 値をリリース(全て 0)します。

▸Auto start on timecode（タイムコードスタート）

タイムコードを受信するとキューリストを実行します。

▸Auto release on timecode lost（タイムコードを消失した時のオートリリース）
タイムコード信号を受け取らなくなった時には、キューリストは自動的に停止します。

▸Mutal Exclude
Allows automatic mutal exclude of cue listings, means, if a cue list with a *mutal exclude ID* greater than zero is started, 自動的に、同じ ID のすべてのキューリストが停止します。

▸Mode Type（モードタイプ）
キューリストの実行モード定義します。通常はリストキューとして設定されており、これを Chaser にすることで、チェイス機能を使用することが出来ます。Chaser は Forward (キューNo.1 から順番)もしくは Backward (最後のキューから順番)の 2 通りの方法があります。また、 Auto を選択するとサウンドアクティブ(音楽同調)となり、ビートに合わせてキューが変化します。

▸Chase Speed / Chase crossfade（チェイススピード／チェイスクロスフェード）
チェイスモードのスピードを設定します。

☝	アドバイス：スピードの値をマウスでダブルクリックするとナビゲーションウィンドウに取り込むことが出来ます。スピードはダブルクリックの速度によって決定します。また、PLAY ボタン（F12）を繰り返し押すと、音楽の BPM 値をミュージックビートに変換することが出来ます。

▸Start next cue / restart cuelist（キューリスト／次のキューの再生）
次のシーンが読み込まれた時や、リストを再生する時のキューリストの再生ボタンを定義します。

▸Release time（リリースタイム）
キューリスト終了の際のフェードタイムを定義します。設定時間をもってキューリストで反映した全ての DMX 値が 0 になります。

▸Speedmaster（スピードマスター）
キューリストのシーケンスをスピードマスターに割り当てると、キューリストの速度を変えることが出来ます（ピッチコントロール）。
Programmer では全部で 10 のスピードマスターが用意されており、キューをスピードによってグルーピングすることが可能になります。スピードマスターは MASTER の下にあります。

▸Extend Submaster Influence（サブマスターの影響を拡張する）
サブマスターの設定はディマー (Intensity)チャンネルではなく、選択したチャンネルグループで行います。

▸Action（アクション）
トリガーマシーンのアクションを入力することが出来ます。キューリストを終了すると、このコマンドが Programmer もしくはメディアエンジンで実行されます。

☝	アドバイス／フェーダーユニットが接続されている時のみトリガーコマンドが有効になります。

8.8. タイムコード

設定されたショーがタイムコードで再生されている時には、タイムコードのレシーバーとしてウィンドウズメディアプレーヤーが使用されています。任意の Audi-CD、MP3、WAV、ビデオなどをソースとして使用することが出来、現在の演出のタイムインフォメーションはタイムコード参照となります。

タイムコードを設定してあるキューは、タイムコードを受け取るとe:cue メディアエンジンで再生を始めます。タイムコードイベントのフリープログラムも可能です（タイムコードショーを設定するには、 キューのプロパティ“をご参照下さい）

8.9. オートメーション

モバイルオペレーション環境においては、Programmer はオーディオとビデオ演出に関する PC の機能を使うことが出来ます。キューもしくはキューリスト（アクション）に設定されている MP3 ファイル、もしくはビデオは、Programmer で演出することが出来ますが、そのためにはフェーダーユニットが接続されていなくてはなりません。

Programmer で作成されたショー（キューとキューリスト）は、外部デバイスに読み込ませることが出来ます。現在は e:cue nano と mediaengine でのご利用が可能です。Nano システムは記憶させたショーの再生のみに機能を絞っている一方、mediaengine は映像や音楽とメディアリンクし、エンターテイメントシステムの中央コンソールとしてメディア間の完全な連係と自動化を図っています。これらのコントロールタスクの管理は、*mediaengine* のトリガー機能で行います。トリガー機能の設定について詳しくは Automation（オートメーション）“をご参照下さい。.

8.10. サウンド to ライト

programmer に含まれているソフトウェアオーディオ DSP ユニットは、トーンデータやミュージックのビート（BPM）を分析、検出することが出来ます。CD やビデオ等の外部ディバイスや、内部マルチメディアプレーヤーを使って入力することが出来ます。
オーディオ DSP には、16 チャンネルのスペクトルアナライザーが含まれています。スペクトルアナライザーの各チャンネルは、どのような灯体のチャンネルへもマッピング(指標化)することが可能です。
また、検出したビートはチェイスの再生速度を操作するために使うことも出来ます。

VIEW/AUDIO DSP もしくは Shift+F9 でオーディオアナライザーがスタートします。

☞	注意：サウンド to ライト機能はフェーダーユニットを接続している時にのみ利用可能となります。

☞	注意： メディアプレーヤーでオーディオを出力させるには、まず Windows のオーディオミキサーがきちんと設定されているか確認して下さい。

F5 もしくはツールバーのアイコン を使ってメディアプレーヤーを再生して下さい。
任意のファイルを選択すると、*オーディオ DAP* が音の分析を始めます。

BPM 値はキューリストの設定によって、チェイサーモード操作となります。

イコライザーのデータは *live-fx* のダイアログで灯体パラメータを置き換えることが出来ます（*live-fx* をご参照下さい）。

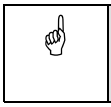
9. Wysiwyg を使って視覚化する

CAST 社の 3D シュミレーションプログラム *WYSIWYG* を使うと、ステージやショーの構成をスクリーンに表示させることが出来ます。よく照明業界では「宿題をする」といわれるように、多様化した舞台シーンに対して効率の良いデザインとライトリハーサルに対する経費の削減手法として、現在はホットトピックとして注目されています。

Programmer は *WYSIWYG* の正規サードパーティであり、完全互換を実現していますので、TCP/IP 経由で DMX データを直接送信することが出来ます。

正しく快適に操作していくためには、まず e:cue のドライバーが、WYSIWYG PC に正しくインストールされていなくてはなりません。

WYSIWYG 内で接続を有効にして下さい。詳しくは WYSIWYG の操作マニュアルをよくお読み下さい。



もし WYSIWYG を e:cueProgrammer と同じ PC でお使いになる場合は、デュアルヘッド用グラフィックカードを使ってモニターを 2 つ用意していただくことを推奨致します。

接続の簡単な説明：

9.1. Step 1: TCP/IP プロトコル

まず最初に、どちらの PC にも TCP/IP プロトコルがインストールされているかご確認下さい。プログラムはこの TCP/IP を介して Wysiwyg にアクセスします。よって、WYSIWYG を同じ PC にインストールしてあるか、違う PC にインストールしてあるかは関係ありません。

次に、もし可能であれば静的 IP アドレス（同じネットワーク内の固定 IP アドレス）をお使い下さい。プライベートネットワークの IP アドレスは、通常 192.168.xxx.xxx で始まります。

IP アドレスを割り当てるのは一度だけにして下さい。

WYSIWYG PC を使うときにもこの IP アドレスが必要になります。

もし *WYSIWYG* と *e:cueProgrammer* を同じ PC でお使いになるのであれば、*デュアルヘッド*グラフィックカードを使ってモニターを 2 つご用意していただくことを推奨します。そうすればマシンの必要メモリがあまり大きくならず、システムにも負担をかけることなく操作出来ます。

9.2. Step 2: Programmer の設定

Programmer をスタートさせ、サブメニューEXTRAS/OPTIONS を開いて下さい。

このメニューでは 3 つのラジオボタンから選択します。

・Wysiwyg is currently not avaivable:
現在 wisywyg と接続がないか、オフラインにしたい時に使用します。
・Wysisyg is running on this computer:
wysiwyg がこのコンピュータで同時に稼動している時にはこのボタンを ON にします。
・Wysiwyg is performing on another computer:
wysiwyg が別の PC で稼動している時に選択します。そのとき、wysiwyg 側の IP アドレスを指定する必要があります。また、念のため双方の PC の IP がユニーク値であること、つまり[DHCP サーバーを参照する]などの設定がなされていないことを確認して下さい。

9.3. Step 3: Wyginstall

TCP/IP による DMX データの伝達は、WYSISYG アプリケーションに e:cue ドライバーが正しくインストールされている時にのみ動作します。Wyginstall は WYSIWYG PC との確実な接続を保証します。

WYSIWYG PC のプログラムを有効にした後で、CAST 社のどのソフトウェアがインストールされたのか示してやらなくてはいけません。

その後でドライバーがインストールされ、必要な変更が施されます。

CAST 社の *WYSIWYG* と *e:cue* の *Programmer* を同じ PC で使用している場合も同じように設定を施して下さい。

9.4. Step 4: WYSIWYG の設定

WYSIWYG メニューから Setup/Request を選択して下さい。
CAST ソフトウェアのコンフィギュレーションにもよりますが、500 から 2000 チャンネルを選択して下さい。 The Request for WYSIWYG was successful というフィードバックが返って来ます。
WYSIWYG メニューから Setup/Device Manager を選択して下さい。ダイアログボックスには現在 CAST もしくは認定サードパーティから発売されているデバイスが表示されます。その中で[e:cue mediaengine]を見つけ、これを選択します。

connected と表示されれば完了です。もし表示されない時には、mediaengine を選択した上で、*connect* ボタンを使って Programmer と WYSIWYG の接続を確定して下さい。

これで connected というステータスが表示されるはずです。このステータスが表示されると e:cueProgrammer と WYSIWYG の間のデータ同期が可能になります。

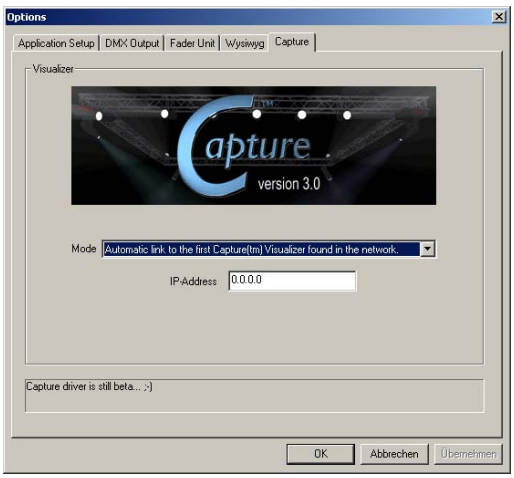
9.5. Step 5: ディバイスをパッチする

WYSIWYG は DMX ポートと DMX ユニバースを区別します。コンフィギュレーションをクリアにするには、ポート 1 を DMX ユニバース A に、ポート 2 を DMX ユニバース B に設定して下さい。

その後、DMX ディバイスにアドレスを設定します。*e:cueProgrammer* との設定に相違がないかご確認下さい。

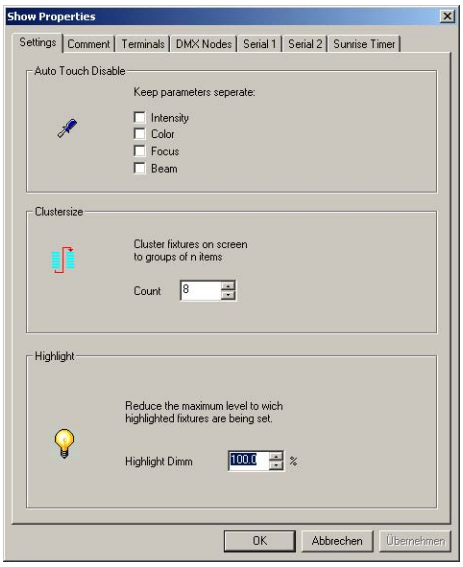
10.Capture を使って視覚化する

e:cue の Programmer は Capture にも対応しています。
日本での発売バージョンに関しては現在実証がなされておりますが、
e:cue ドイツホームページ経由で Capture ソフトウェアのサンプル版を
ダウンロードして使用することも可能です。
詳しい設定の方法は e:cue ウェブサイトをご参照下さい。
www.ecue.tv



11.設定

11.1.ショープロパティ



SHOW PROPERTIES のダイアログで施された設定は、現在のショーに保存されます。

▸ ハイライトディマー(Highlight Dimmer)
ハイライト機能で選択した灯体の明るさ最高値を定義します（7.5 をご参照下さい）。

▸ パラメーターを個別にキープする(Keep Parameters Separate)
ここでのチェックボックスを ON にすると、programmer モードでデザインする時、関連チャンネルのブロックが外れ、個別に操作することになります。ライブラリーに登録している全てのインテリジェントライトは主にこの 4 つのグループにチャンネルが分かれており、各々のグループのうちのどれかの 1 チャンネルを選択すると属するグループ全てのチャンネルに対して変更したり Touch/Untouch の選択ができますが、このチェックボックスを ON にすることにより、個別に操作することになります。

▸ Cluster Size（クラスターサイズ）
スクリーンビューの同じ種類の複数灯体の定義は、ひとつのヘッドにまとめられます。これはプレビューウィンドウでも有効です。

▸ User Comment（ユーザーコメント）

ショーが読み込まれている間ユーザーに表示されます。

▸ Terminals(端末)

e:com ターミナルをネットワークに登録して、プログラム可能なオペレーションターミナルとして *mediaengine* で使用することが出来ます。

▸ DMX Nodes(DMX ノード)

e:node のイーサネットを DMX ゲートウェイへネットワーク内でレジスターすることが出来ます。*メディアエンジン*や *Programmer* は DMX データを特定のアドレスへ送信します。

▸ Serial1,Serial2(シリアル 1、シリアル 2)

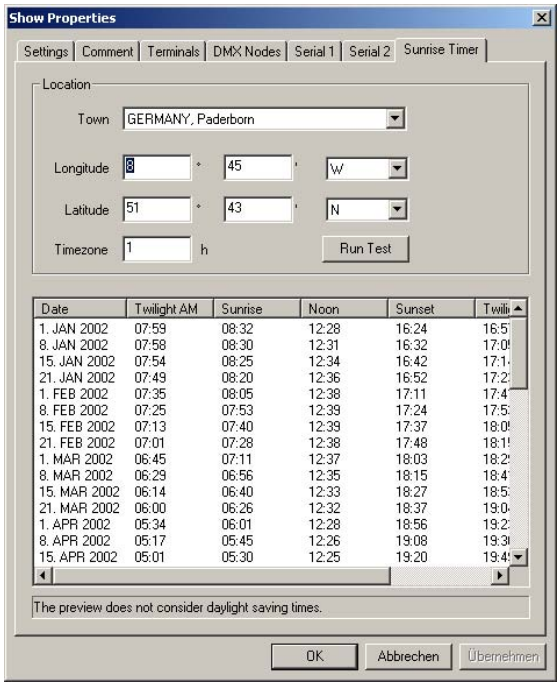
メディアエンジンのシリアルインターフェイスパラメーターは Programmer で直接定義することが出来ますが、メニュー *Engine/ External Ports* で変更することも出来ます。 ショーが *mediaengine* に読み込まれると、これらの設定が有効になります。

▸ Sunset Timer(日の出タイマー)

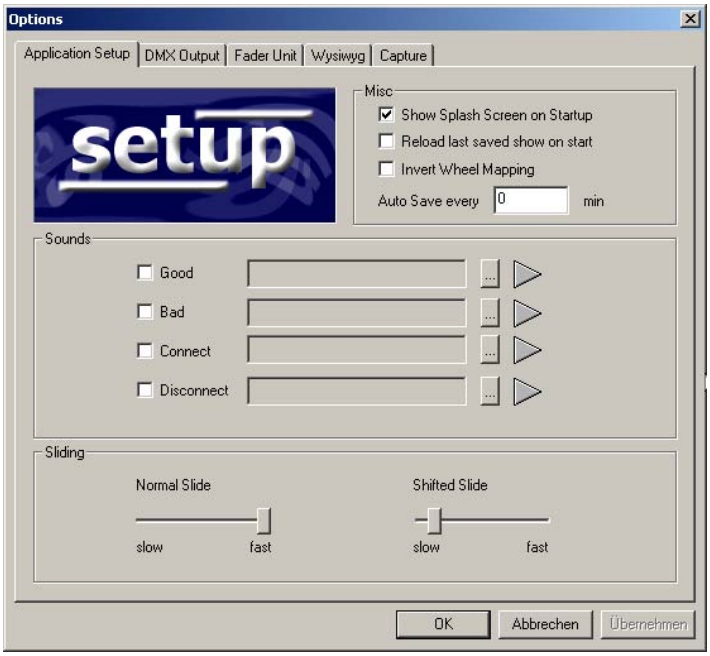
*メディアエンジン*をお使いいただくと、日の入／日の出時間が設定出来ます。時間はディバイスを設置する場所によって変わります。変わります。*メディアエンジン*のシリアルインターフェイスのパラメーターは Programmer で直接定義することが出来ます。メニュー Enging/External を使って変更を施すことも出来ます。これらの設定は、ショーが*メディアエンジン*に読み込まれて初めて有効になります。

世界都市リストの中から*メディアエンジン*を設置する場所を選択して下さい。*Run Test* で時間が算出されます。

自動プログラムでは、これらの時間はトリガーイベントとして利用可能です。


Show Properties
Settings
Comment
Terminals
DMX Nodes
Serial 1
Serial 2
Sunrise Timer
Location
Town
GERMANY, Paderborn
Longitude
8
45
W
Latitude
51
43
N
Timezone
1
h
Run Test
Date
Twilight AM
Sunrise
Noon
Sunset
1. JAN 2002 07:59 08:32 12:28 16:24
8. JAN 2002 07:58 08:30 12:31 16:32
15. JAN 2002 07:54 08:25 12:34 16:42
21. JAN 2002 07:49 08:20 12:36 16:52
1. FEB 2002 07:35 08:05 12:38 17:11
8. FEB 2002 07:25 07:53 12:39 17:24
15. FEB 2002 07:13 07:40 12:39 17:37
21. FEB 2002 07:01 07:28 12:38 17:48
1. MAR 2002 06:45 07:11 12:37 18:03
8. MAR 2002 06:29 06:56 12:35 18:15
15. MAR 2002 06:14 06:40 12:33 18:27
21. MAR 2002 06:00 06:26 12:32 18:37
1. APR 2002 05:34 06:01 12:28 18:56
8. APR 2002 05:17 05:45 12:26 19:08
15. APR 2002 05:01 05:30 12:25 19:20
The preview does not consider daylight saving times.
OK
Abbrechen
Übernehmen

11.2.オプション



Application Setup(アプリケーションセットアップ)は、アプリケーションの基礎部分を定義するためのメニューです。従って、ここでの設定はアプリケーションの全ての動作に影響をおよぼすことに注意して下さい。を使用するには、グローバルな設定がプログラムに必要であると定義して下さい。プログラムでアクションの設定が執り行われた任意のサウンドは、Programmer を使用する際にスタートスクリーンを有効にするべきか否か定義することが出来ます。

・Show Splash Screen on Startup
プログラムをスタートする際、e:cue ロゴの表示をするかどうかを決定します。
・Reload last saved show on start
前回使用したショープログラムを自動的に呼び出します。
・Invert Wheel Mapping
フェーダーユニット使用時、エンコーダーホイールの回転を逆にします。（反時計方向で値が増加します。）
・sounds

PC から送られる様々なリアクション(反応)に対する音を定義します。
・Sliding★重要★この設定は必ず必要です！
マウスやトラックボールでのパラメータ操作に対し、スピードを決定します。特に Shifted Slide は 16 ビット灯体のパン／チルトを操作する時に必要です。お好みの値を決めます。

12.オートメーション

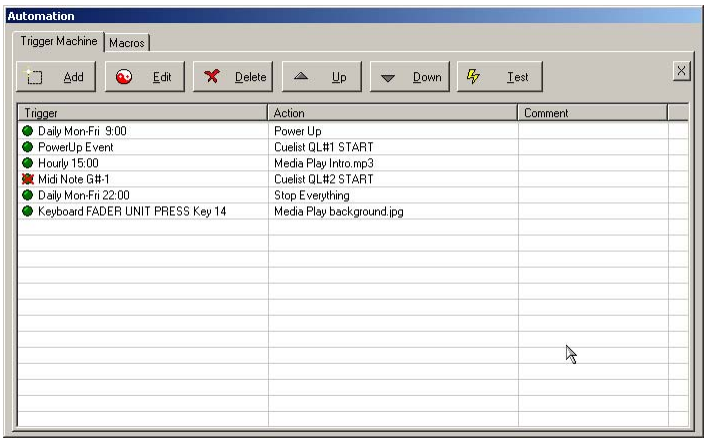
*ENGINE/AUTOMATION*で、*メディアエンジン*のオートメーションを設定することが出来ます。

12.1.トリガーマシン

トリガーマシン を使うと簡単にルールの設定を行うことが出来ます。この設定は後に *mediaengine* で使います。

使用例：

- mediaengine の電源を、毎週日 8:10 にオンにする
- キューリスト 1 を毎時 5 分に再生する
- MIDI ノート C#2 を受信した再荷はキューリスト 2 を再生する
- キーを押すとキューリスト 15 と MP3　jingle.mp3 が再生する
  など、さまざまなコマンドを組み合わせることによりほぼ完璧に近いまでのオートメーションプレイが実現します。



※この機能を使用するには Mediaengine が必須です！

*トリガーマシーン*で行われた各入力は個々に有効になります。つまり、表示された順番に反応するではなく、すべての入力は同じように取り扱われると言うことです。

上の事例で言えば、ディバイスは月曜から金曜日の 9:00 になるとキューリスト QL1 の再生を始めます。15 分後にサウンドファイル

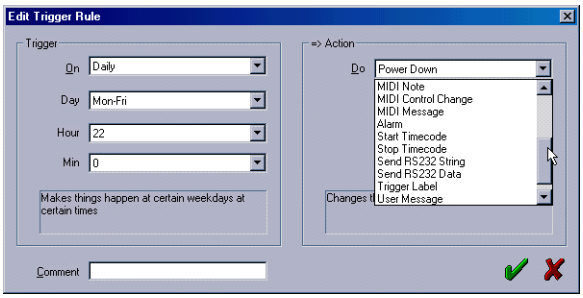
Intro.mp3 が再生されます。午後 10:00 になると各出力は停止となり、ディバイスの電源は自動的にオフとなる、というオートメーションを割きに作っていたとして、後で MIDI 鍵盤を mediaengine に接続し、G#1 を叩いたときに QL2 がスタートするように設定したとします。各々のルールは独立して働いているので、後から定義したルールは限定条件をつけない限り、いつでも有効になります。また、一時的にストップすることも削除することも可能ですが、それぞれのトリガールールに何らかの関連オプションをつけていない限りは、他のルールに影響をおよぼすことはありません。

(例)Mediaengine と Faderunit をつなぎ、接続されている *Faderunit* のキーを使って、ビデオ出力から background.jpg を出力する、というプログラムを仮定します。ここでは、

何を	いつ	どうすれば	どうなる

を定義すれば良いので、
何をーMediaengine につながっている Faderunit のあるキーを
いつーいつでも
どうすればー押すと
どうなるー background.jpg を mediaengine から VGA で出力

となります。このような定義を「トリガールール」と称し、下図のダイアログボックスで行います。



これを *IF　SO Rules*（IF　SO ルール）と呼び、常に定義されたトリガーから再生され、定義されたアクションがこれに従います。

トリガー：

Power Up	ディバイスのスイッチをオンにする
Daily	毎日
Hourly	毎時
Cuelist	キューリストの再生、もしくは停止
Keyboard	キーボード入力 (*)
Midi Note	MIDI ノートメッセージ
Timecode	タイムコードの受信（SMPTE or MTC）
Label	トリガーグループ (**)
Sun Event	（予め設定された）日の出／日の入時間
Every Second	毎秒
Terminal Event	ネットワーク内の *e:com* ターミナルのキーをプレスする

(*)キーボードの入力は：　メディアエンジンのフロントパネルのキー、もしくは接続してあるキーボード、もしくは SPS 入力で執り行って下さい。

(**) 特定のトリガーを組み合わせて新しいラベルを定義することができます。(インターレースプログラム)

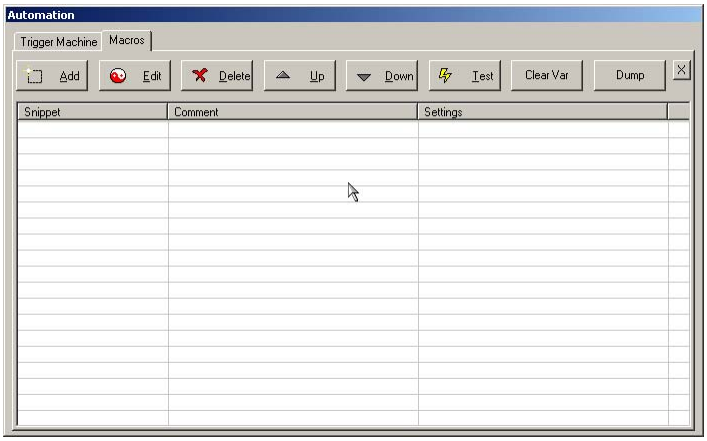
アクション：

Power Up	デバイスのスイッチをオンにする
Cuelist	キューリストを再生(もしくは停止)する
Stop Everything	全てのプレイバックを停止する (DMX、オーディオ、ビデオ)
Media Play	メディアファイルを再生
Media Stop	メディアファイル（オーディオ、ビデオ）停止
Media Eject	Ejects medium （外部ディバイスのみ)
Set Output up	SPS 出力信号を *high* に設定する
Set Output down	SPS 出力信号を *low* に設定する
MIDI Note	MIDI ノートを送る

MIDI Control Change	MIDI メッセージを送る
MIDI Message	MIDI を使って任意のストリングを送る
Start Timecode	内部 SMPTE タイムコードを再生する
Stop Timecode	内部 SMPTE タイムコードの出力を停止する
Send RS232 String	指定した COM ポートに RS232 ストリングを送信する
Send RS232 Data	RS232 を使ってデータフローを送る
Trigger Label	定義したラベルを有効にする
User Message	メディアエンジンのディスプレイにテストメッセージを表示する（logbook）
Exec Command	マクロ言語のコマンドを直接実行する
Call Snippet	マクロを有効にする
Power down	デバイスのスイッチをオフにする

12.2. マクロ

より複雑なシーケンスをプログラムするために、*mediaengine* ではマクロ言語を使用することが出来ます。



このマクロ言語は、通常コンピュータ業界でもっとも標準とされる C 言語を使用して行われ、カスタマイズできるコマンドラインやもっとも分かりやすい高級言語が魅力の言語です。e:cue は世界中のライティングデザイナーやプログラマーとの共同作業を実現するために、この「C」を積極的に採用しています。
　現在日本においても数々のマクロプログラム事例がこの C によるマクロコマンドラインによって作成され、トータルコストの削減に大きな威力を発揮しています。

例：
ターミナルナンバー2 の 4 キーをプレスするとキューリストが再生されます。パラレルメッセージはネットワークターミナルのディスプレイに表示されます。任意の端末の 3 キーが押された時には、定義された再生キューリストの範囲はブラインドアウトとなります。

アップデートの繊細については、オンラインサポートをご参照下さい（F1）。

Version 3.0.181

12.2.1. Variables

Variables can be of two types: 'int' or 'string'.
Integer variables can also be defined as unidimensial arrays (see below).

If a variable name starts with an underline (_), it will be handled 'global',
that means it persists even when the snippet has ended.

This is a great advantage, if you want to keep track about the state of something
longer than the usual quite short runtime of a code snippet.
But there's a little hook, that makes handling of global variables a bit tricky:
If you start a snippet, that created a global variable, a second time, you'll get
an error when the snippet reaches the line, where the variable was declared the first time.
To handle this situation, you may query the existence of a variable and, if it already
exists, skip it's re-declaration. The function 'exists()' returns true, if the given
variable already exists.

Example:

```
if (!exists(_hello))
int _hello;
```

12.2.2. int

int var1, var2, ...;

Declare variables of type integer.

If the variable name starts with an underline (_), the variable is declared 'global' (see above).

12.2.3. int arrayname[size]

int var1[size1], var2[size2], ...;

Declare variables as arrays of integer.
If the variable name starts with an underline (_), the variable is declared 'global' (see above).

12.2.4. string

string var1, var2, ...;

Declare variables of type string.
Strings have a maximum length of 256 bytes.
Arrays of strings are currently not allowed.
If the variable name starts with an underline (_), the variable is declared 'global' (see above).

12.2.5. if

```
if (condition) command1;
else
command2;
```

The command1 will be executed, when the condition is true, i.e. not zero.
The command2 will be executed, when the condition is false.
The 'else' path is of course optional.
If you want to execute more than one statement in dependence of a condition, you
have to use curly brackets, like this:

```
if (a==1)
{
printf("a is now 1\n");
printf("It's not 2\n");
}
```

12.2.6. for (init; condition; increment) command;

The for keyword executes statement repeatedly.

First, the initialization (init) is evaluated. Then, while the conditional expression (condition) evaluates to a nonzero value, statement is executed and the loop expression (loop-expr) is evaluated. When condition becomes FALSE, control passes to the sta

Example:
var i;
for (i=0; i<10; ++i)
printf("i is now %d\ n",i)

12.2.7. return

return value;

Return from a function and return a value

12.2.8. switch, case, break

switch(expression)
{
[case constant-expression:]
. . .
[statement]
. . .
[default:
statement]
}
The switch and case keywords evaluate expression and execute any statement associated with constant-expression whose value matches the initial expression. If there is no match with a constant expression, the statement associated with the default keyword is executed. If the default keyword is not used, control passes to the statement following the switch block. For related information, see if and break.
Examples

All three statements of the switch body in the following example are executed if c is equal to 'A' since a break statement does not appear before the following case. Execution control is transferred to the first statement (capa++;) and continues in order

```
switch( c )
{
case 'A':
capa++;
case 'a':
lettera++;
default :
total++;
}
```

In the following example, a break statement follows each statement of the switch body. The break statement forces an exit from the statement body after one statement is executed. If it is equal to –1, only n is incremented. The break following the statement

```
switch( i )
{
case -1:
n++;
break;
case 0 :
z++;
break;
case 1 :
p++;
break;
}
```

12.2.9. while

```
while( expression )
statement
```

The while keyword executes statement repeatedly until expression becomes 0.

12.2.10. exit

12.2.11. Binary Operators

<= less or same
>= greater or same
== equal
!= not equal
<< logic shift left
>> logic shift right
&& boolean and
¦¦ boolean or
%% modulo
+ add
- subtract
* multiply
/ divide
& logic and
¦ logic or
^ logic xor
< less than
> greater than

12.2.12. Assignment Operators

= let
%%= let modulo
+= let add
-= let subtract
*= let multiply
/= let divide
&= let logic and
¦= let logic or
^= let logic xor

12.2.13. String Operators

= copy one string to another
+ cat two strings in an expression
+= cat two strings

12.2.14. Syntax

Snippets usually run for a very short amount of time. A typical runtime length should be
articulately below 0.1 sec. That means, if you want to realize more complex applications,
you should create them as a kind of state machine. This state machine is always being
triggered again by the trigger machine, whenever something happens, the may be of interest
for your application.

The application does not necessarily have to be implemented in one snippet.
Rather it is useful to take a number of snippets, for each special event.

12.2.15. clear

Deletes all global variables.

12.2.16. printf

int printf(string format, ...);

Output a standard message to the logbook.

Return value: Always zero

12.2.17. alert

int alert(string format, ...);

Output a alert message to the logbook.

Return value: Always zero

12.2.18. abs

int abs(int value);

Returns the absolute value of a number.

12.2.19. check

int check(int par1,int par2,int par3);

Return par2 if par1 is true i.e. non equal zero,
otherwise return par3.

12.2.20. random

int random(int range);

Returns a random number (0..range-1).

12.2.21. strlen

int strlen(string txt);

Returns the length of a string.

12.2.22. val

int val(string txt);

Converts a string to a integer value.

12.2.23. strcmp

int strcmp(string txt1,string txt2);

The return indicates the lexicographic relation of
string1 to string2.
<0 string1 less than string2
0 string1 identical to string2
>0 string1 greater than string2

12.2.24. stricmp

int stricmp(string txt1,string txt2);

The return indicates the lexicographic relation of string1 to string2.
<0 string1 less than string2
0 string1 identical to string2
>0 string1 greater than string2

In opposite to strcmp this function is not case sensitive

12.2.25. getchar

int getchar(string source,int n);

Returns the signed Ascii-Code of the n-th char in a string(-128...127).

12.2.26. setchar

int setchar(stringptr dest,int n ,int ch);

Sets the n-th char in a string to the given signed value (-128...127).

Return value: Always zero

12.2.27. getuchar

int getuchar(string source,int n);

Returns the unsigned Ascii-Code of the n-th char in a string (0...255).

12.2.28. setuchar

int setuchar(stringptr dest,int n,int ch);

Sets the n-th char in a string to the given signed value (0...255).

Return value: Always zero

12.2.29. getarg

int getarg(int arg_id);

Returns an argument for the snippet
.When the snippet is invoked by a trigger, you can find out the exact trigger
values with this function:

Values for getarg(0):
0: POWER_UP
1: WEEKDAY
2: HOURLY hour, second
3: CUELIST_EVENT
4: KEY_EVENT
5: MIDI_NOTE
6: SMPTE_EVENT
7: LABEL
8: SUNRISE (never happens)
9: TIMER_TICK (never happens)
10: TERMINAL_EVENT term_id, event_id, key_id

12.2.30. midstr

string midstr(string src,int pos,int count);

Returns a partial string
'pos' is the zero based start position.
'count' is the maximum number of chars, that will be returned

12.2.31. createstring

string createstring(int char,int n);

Creates a string, that contains n times the given char.

12.2.32. format

string format(string format, ...);

Returns a string. The format is like at the printf function.

12.2.33. exists

int exists(labelname name);

Returns True, if the variable exists.

12.2.34. undefined

int undefined(labelname name);

Returns True, if the variable *not* exists.

12.2.35. size

int size(labelname name);

Returns the size of the array, when the array exists
Returns -1 if no variable with that name exists
Returns 0 if the variable isn't an array

12.2.36. GetTerminalCount

int GetTerminalCount();

Returns the number of registred terminals.

12.2.37. TerminalSend

int TerminalSend(int term_id,int line_id,string message, ...);

Send a message to a terminal.
Using Zero as terminal number sends a broadcast to all terminals.
If line_id is zero, the terminal will be cleared, before displaying

the message.

Return value: Always zero

12.2.38. TerminalSetup

int TerminalSetup(int term_id,int item_id,string caption, ...);

Setup an item on a terminal.
Using Zero as terminal number sends a broadcast to all terminals.
Items are for example the buttons on a terminal. You may use this
this command, to give each button it's custom caption
After the use of this command, the addressed terminals stop sending
initialisation messages.

Return value: Always zero

12.2.39. GetCuelistStatus

string GetCuelistStatus(int cuelist_id,int par);

Get the status of the given cuelist.
'id' is the zero based number of the cuelist.
The parameter 'par' selects the status field.
0: Play Status
1: Extended List Name
2: Submaster Position
3: Short List Name

Return value: Status

12.2.40. GetCuelistName

string GetCuelistName(int cuelist_id);

Get the name of the given cuelist

12.2.41. GetMutexStatus

int GetMutexStatus(int mutex_code);

If one cuelist of the mutal exclude group is running, the zero based.
cue number will be returned. Otherwise a value below zero will be
returned.

12.2.42. MutexRelease

int MutexRelease(int mutex_code);

If one cuelist of the mutal exclude group is running, that cuelist
will be stopped.

12.2.43. StartCuelist

int StartCuelist(int cuelist_id);

Start a cuelist
'id' is the zero based number of the cuelist.

Return value: Always zero

12.2.44. StopCuelist

int StopCuelist(int cuelist_id);

Stop a cuelist
'id' is the zero based number of the cuelist.

Return value: Always zero

12.2.45. GetCurrentCue

int GetCurrentCue(int cuelist_id);

Returns the zero based index of the current cue if the cuelist

is running.
When the cuelist is idle, the return value is -1.

12.2.46. ReleaseAll

int ReleaseAll();

Stop all cuelists

Return value: Always zero

12.2.47. PowerUp

int PowerUp();

Power on the media engine

Return value: Always zero

12.2.48. PowerDown

int PowerDown();

Set the media engine to standby

Return value: Always zero

12.2.49. MediaPlay

int MediaPlay(int term_id,string titel);

Replay Media on given terminal.
Terminal 0 stands for media engine.

Return value: Always zero

12.2.50. MediaStop

int MediaStop(int term_id,string titel);

Stop Media Playback on given terminal.
Terminal 0 stands for media engine.

Return value: Always zero

12.2.51. MediaEject

int MediaEject(int term_id);

Eject Media on given Terminal.

Return value: Always zero

12.2.52. SetOutput

int SetOutput(int output_id);

Set the given output on the media engine to ON
'output_id' is zero based.

Return value: Always zero

12.2.53. ClrOutput

int ClrOutput(int);

Set the given output on the media engine to off

Return value: Always zero

12.2.54. GetInput

int GetInput(int);

Return the value of the given input on the media engine.

Return value: 1 if input is high

12.2.55. MidiNoteOn

int MidiNoteOn(int channel,int note,int velocity);

Channel: 0..15

Note: 0..127
Velocity: 0..127
Send a midi note on message.

Return value: Always zero

12.2.56. MidiNoteOff

int MidiNoteOff(int channel,int note,int velocity);

Send a midi note off message.
Channel: 0..15
Note: 0..127
Velocity: 0..127

Return value: Always zero

12.2.57. SendMidi

int SendMidi(int byte);

Send a single unsigned byte via midi (0..255)

Return value: Always zero

12.2.58. TimecodeStart

int TimecodeStart(int hh,int mm,int ss,int ff);

Start Timecode Generator at given time offset.

Return value: Always zero

12.2.59. TimecodeStop

int TimecodeStop();

Stop the Timecode Generator.

Return value: Always zero

12.2.60. SendString

int SendString(int id,string message);

Send a string via a RS232 interface
'id' zero based index of the serial port

Return value: Always zero

12.2.61. SendBinary

int SendBinary(int id,int message,int length);

Sends binary data via RS232 interface
'id' zero based index of the serial port
To setup the binary data, declare a string and
set the single bytes via 'setuchar' (see above).
'length' is limited to a maximum of 256 bytes.

Return value: Always zero

12.2.62. GetGrandMaster

int GetGrandMaster();

Return the curent state of the grandmaster (0...4096).

12.2.63. SetGrandMaster

int SetGrandMaster(int val);

Set the grandmaster to the given Value (0...4096).

Return value: Always zero

12.2.64. GetSubMaster

int GetSubMaster(int id);

Return the curent state of the given submaster
(0...4096)

12.2.65. SetSubMaster

int SetSubMaster(int id,int value);

Set the given submaster to the given value (0...4096).

Return value: Always zero

12.2.66. GetSpeedMaster

int GetSpeedMaster(int id);

Return the curent state of the given speedmaster (0...10000).

12.2.67. SetSpeedMaster

int SetSpeedMaster(int id,int value);

Set the given speedmaster to the given value (0...10000)
100%% is equal to a value of 3162.

Return value: Always zero

12.2.68. SetSectionRights

int SetSectionRights(int rights,int start_list,int count);

Setup section access rights to the given cuelists.

'rights' is a binary coded mask, in which each of the first eight bits correspons
to one of the eight user defined sections.

'start_list' is the zero based number of the first cuelist, which should be set up

'count' is the number of cuelists, the should be setup

Return value: Always zero

12.2.69. GetSectionRights

int GetSectionRights(int cuelist);

Return section access rights for the given cuelist (see above).
'id' is the zero based number of the cuelist.

12.2.70. Call

int Call(string name,int arg1,int arg2,int arg3);

Call another snippet. 3 Arguments can be given to the snippet as
parameters. The called snipped can access these parameters via the
function 'getarg()'.
The maximum recursion deep is limited to a deep of MAX_RECURSION_DEEP.

12.2.71. NiceTime

string NiceTime(string format);

Return the system time as string.
The format parameter descripes the, how to set up the time as string:
%a Abbreviated weekday name
%A Full weekday name
%b Abbreviated month name
%B Full month name
%c Date and time representation appropriate for locale
%d Day of month as decimal number (01 – 31)
%H Hour in 24-hour format (00 – 23)
%I Hour in 12-hour format (01 – 12)
%j Day of year as decimal number (001 – 366)

%m Month as decimal number (01 – 12)
%M Minute as decimal number (00 – 59)
%p Current locale, s A.M./P.M. indicator for 12-hour clock
%S Second as decimal number (00 – 59)
%U Week of year as decimal number, with Sunday as first day of week (00 – 53)
%w Weekday as decimal number (0 – 6; Sunday is 0)
%W Week of year as decimal number, with Monday as first day of week (00 – 53)
%x Date representation for current locale
%X Time representation for current locale
%y Year without century, as decimal number (00 – 99)
%Y Year with century, as decimal number
%z, %Z Time-zone name or abbreviation; no characters if time zone is unknown
%% Percent sign

12.2.72. CopyCuelist

int CopyCuelist(int cuelist);

Copy a cuelist into the clipboard.
Remark: Works *not* on the media engine.

12.2.73. PasteCuelist

int PasteCuelist(int cuelist);

Paste a cuelist from the clipboard.
Remark: Works *not* on the media engine.

12.2.74. SetMutexID

int SetMutexID(int cuelist,int value);

Sets the mutex code for the given cuelist to the given value.

13.フェーダーユニット

e:cue フェーダーユニットをお使いいただくとプログラミングが簡単になります。13 のフェーダーと 5 つのジョグダイアルで *e:cueProgrammer* をより直感的に、素早く操作することが出来ます。

13.1. フェーダーユニットの設置と初期設定

フェーダーユニットにはシリアルケーブルとパワーサプライユニットが付属しています。この付属ケーブルをなくされた時には、新しいものを注文して下さい（16.2 をご参照下さい）。

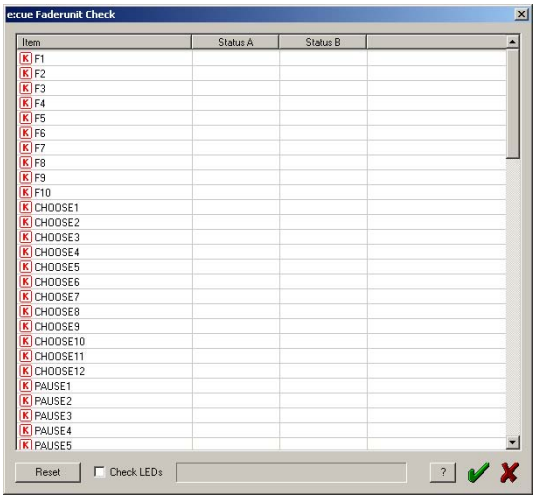
付属のシリアルケーブルを使って PC とフェーダーユニットを繋いで下さい。付属のパワーサプライユニットを使って電源を供給して下さい。

本体上部左側にあるパワーLED が点滅を始めます。
EXTRAS/OPTIONS を使って Programmer との接続が確定すると、点滅は点灯になります（5.1 をご参照下さい）。また、programmerPC 画面左側のステータスモニターにも Fader Unit Online の表示が出ます。
もしもきちんと動作しない時には、まずケーブルをご確認下さい。

それでも作動しない時には、メニュー[EXTRAS/CHCK FADERUNIT] を使ってディバイスをテストして下さい。

次にディバイスのすべてのキーを押し、ジョグダイアルを 3 度回しフェーダーを動かして下さい。もし選択したアイテムがきちんと動作すれば、LED は緑に点灯します。

注意：LED が破損していないかどうかもご確認下さい。


ecue Faderunit Check
Item
Status A
Status B
F1
F2
F3
F4
F5
F6
F7
F8
F9
F10
CHOOSE1
CHOOSE2
CHOOSE3
CHOOSE4
CHOOSE5
CHOOSE6
CHOOSE7
CHOOSE8
CHOOSE9
CHOOSE10
CHOOSE11
CHOOSE12
PAUSE1
PAUSE2
PAUSE3
PAUSE4
PAUSE5
Reset
Check LEDs
?

14.メディアエンジン

14.1.概要

mediaengine, Nano, Twilight などの DMX 出力ディバイスは *Programmer* ソフトウェアを使って操作しますが、中でも*メディアエンジン*は他のディバイスよりもよりパワフルにその威力を発揮します。メディアエンジンは、そのなかでも DMX ショーの再生のみならず、ユニバーサルメディアコントロールを前提としたトータルソリューションプレイバックユニットとして位置付けられています。

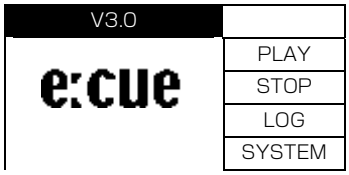
通常、この mediaengine は主に設備ユースで使用されることが多いですが、卓越したマルチメディアソリューションを実現するため、ライブでの使用も前提にして作られています。
Mediaengine と Programmer PC をイーサネットで接続し、オンラインするだけで PC 上のデータが直接反映され、しかも mediaengine がもつ全ての機能を直接使用することが可能です。

また、PC（ノートブック）と *mediaengine* をオプションの W/L コネクションキットでワイヤレスで接続することにより、ケーブルがなくても、直接ステージへプログラムを送出することが可能になります。ステージのどこからでもプログラミングが可能になる画期的なシステムが実現します。

メディアエンジンの正面部分には 6 つのキーとジョグダイアルが配列されています。これを使ってドライバのステータスを呼び出すことも出来ます。また進行中のショーに入り込むことも出来ます。その仕様は後ほど詳しく説明します。

14.2.メディアエンジンのメインメニュー

電源を入れると、メディアエンジンのメインメニューが表示されます。通常は e:cue のロゴが表示されています。キューリストや MP3 などのアクションがメディアエンジンで再生されはじめると、これがメインメニューに表示されます：



Media Engine ベーシックモード

メディアエンジンが Programmer に接続されると、Programmer 側 PC の IP アドレスがメインメニューに表示されます：

Media Engine オンラインモード

メインメニューからサブメニューに変換することも出来ます。メインメニューに戻るには常に ESC を使って下さい。

14.2.1. サブメニュー再生

サブメニューPlay でディスプレイに一番初めのキューリストの名前が表示されます。ソフトキーには *Play*, *Pause*, *Stop* and *Level* が表示されます。

The Submenu [Play]（サブメニュー/再生）

- *Play* でキューリストの再生、またはリストの次のキューに移行します
- *Pause* で再生の一時停止、またはひとつ前のキューに移行します
- *Stop* でキューリストのプレイバックは終了します

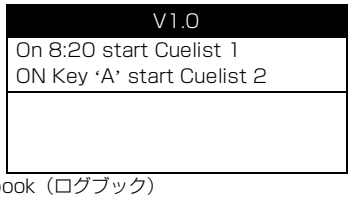
Level を使うとキューリストのサブフェーダーに変更を及ぼすことが出来ます。*Level* を押すとディスプレイ左側のバーが点滅を始めます。サブフェーダーの値はジョグを回して変更して下さい。*Level* を再度押すと、プログレスの設定を停止します。

他のキューリストを選択する時には、ジョグダイアルをまわして下さい。

ESC でメインメニューに戻ります。

14.2.2. サブメニューログ

アクションとイベントはメディアエンジンのインターナルログブックに取り込まれます：



The Logbook（ログブック）

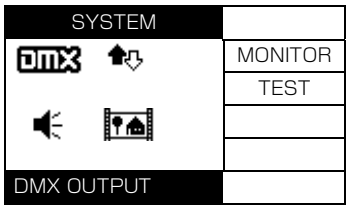
ログブックをスクロールさせる時にはジョグダイアルを使って下さい。
ソフトキー3 でログブックを削除することが出来ます。

14.3.メディアエンジンのシステムメニュー

システムメニューはより多くの機能のために多数のアイコンを用意しています：

DMX	DMX 出力とテスト
	I/O インターフェイス
	オーディオアウト (MP3)
	VGA 出力 (JPG)
	MIDI モニター
SMPTE	SMPTE タイムコード
	システム時計
	フロッピーアクセス
	エキストラ
	パスワードを使ってロック／アンロック
	デバイス設定

ジョグダイアルでアイコンを選択して下さい。ソフトキーの横に使用可能なアクションが 4 つ挿入されます：



システムメニュー

14.3.1. DMX / モニター

DMX このスクリーンを使って現在の DMX 出力をディスプレイに表示させることが出来ます。ジョグダイアルを使って、1DMX ユニバース（512 チャンネル）をスクロールさせることが出来ます。ソフトキー0 を使うと DMX ユニバース A から H へ変換出来ます。

14.3.2. DMX / テスト

DMX DMX テストは DMX 出力へのインタフェアを可能にします。ソフトキー 1 で *Curr*、*Row*、*All* モードへ変換します。
ソフトキー2 を使うとモード *Move* と *Modify* を切り替えます。

ムーブモードでは、ジョグダイアルを使って 1DMX ユニバース（512 チャンネル）をスクロールさせることが出来ます。ソフトキー0 を使うと、DMX ユニバース A から H へ変換出来ます。

モディファイモードで現在の DMX チャンネル（Curr）、ライン（Row）、もしくは現在のユニバース（All）のすべてチャンネルの DMX 出力を変更します。

14.3.3. PLC／ビュー

SPS とも呼ばれ、6 つのメイク接点トリガー(光アイソレート)と 2 つのタリー出力(+12V)からなるトリガー入出力です。単純なメイク接点の入出力リレーで、さまざまな機器のリモートなどにお使いいただけます。このサブメニューを使って、ポテンシャルフリーインプットのレートを表示することが出来ます。出力のレートは *トグル*ボタンを使って変更して下さい。

注意：キューリストはバックグラウンドやトリガーマシーンのルール設定の際にも動作しており、出力のステータスにも影響を及ぼします。

14.3.4. サウンド／再生

予め FTP 経由でアップデートされた MP3 ファイルを外部デバイス無しで mediaengine から再生することが出来ます。MP3 ファイルは mediaengine 内のハードディスクに格納されており、いつでも取り出せて使用することが出来ます。すべてのイベントにおいてオーディオファイルのプレイバックが可能です。時間／キープレス／キュー／MIDI ノート／タイムカードなど、出力のきっかけをつくるトリガーは何でも構いません。サウンドモジュールは自由に設定することが出来ます。
すべてのシステムをオーディオファイルに保存、挿入します。ソフトキー0 をプレスするとオーディオの演出が再生されます。

14.3.5. サウンド／停止

オーディオのプレイバックを停止します。

14.3.6. メディア／再生

メディアエンジンの VGA 出力は、スライドショー（JPG フォーマット）にも使用することが出来ます。もしメディアエンジンをビーマーと一緒にお使いになれば、インタラクティブな DIA プロジェクターを創ることが出来ます。外部 DVD プレーヤーを使って、ショーと同期させることも出来ます。

VGA 出力で画像を出力させることが出来ます。画像だけでなくオーディオデータを使って自由にプログラムを組むことが出来ます。各トリガーに画像を出力させることも可能です。

すべてのイメージファイルがシステムに保存されます。ソフトキー0 を押すとイメージファイルのプレイバックが再生されます。
※2003 年 1 月現在では、jpeg の他 mpeg 再生のためのバージョンアップが行われようとしています。詳しくはメーカー情報をご覧ください。

14.3.7. メディア／停止

イメージファイルのプレイバックを停止します。VGA 出力に e:cue ロゴが表示されます。

14.3.8. Midi／ビュー

受信した midi イベントをすべて表示します。ジョグダイアルを使ってリスト内をスクロールさせることが出来ます。ソフトキー0 でリストを削除します。

14.3.9. SMPTE／ビュー

SMPTE タイムコードユニットのステータスを表示する

アドバイス：SMPTE タイムコードモードは mediaengine には存在していません。もし定かでなければ、お近くのディーラーまでお問い合わせ下さい。

14.3.10. オプション／Misc

ディバイスの設定を変更するために使います。ここで施された設定はディバイスに固定され、現在アップロードされているショーとは区別して取り扱われます。

Zone	時間と日付けの設定をディスプレイに挿入するフォーマットを定義する
SMPTE	タイムコードの出力フォーマットを定義する
S1, S2, S3	タイムコードを再生する時間を定義する。この設定は、タイムコードがディスプレイに直接再生された時に使用されます。

14.3.11. オプション／Net

ネットワークプロトコル TCP/IP に関する重要な設定はここで行います。

IP アドレス	IP アドレス（デフォルト設定は 192.168.123.1）
サブネットマスク	サブネットマスク（通常は 255.255.255.0 もしくは 255.255.0.0 となっていますが、システムの管理者にまず確認して下さい）
ゲートウェイ	ローカルネットワークのデフォルトゲートウェイ。もしお使いのネットワークにデフォルトのゲートウェイがなければ IP アドレスで設定して下さい。しかし最後のドットの後の数字は 254 であるべきです。 例えば IP=192.168.123.1 のゲートウェイアドレスは 192.168.123.255 となります。
DHCP クライアント	設定が YES にセットされている時には、メディアエンジンはローカルネットワーク上の DHCP サーバーから IP アドレス、サブネットマスク、ゲートウェイを読み込みます。もしこれらの設定が変更された時には、メディアエンジンも再起動する必要があります。

PC に e:cueProgrammer ソフトウェアと TCP/IP プロトコルがインストールしてあるか確認して下さい。e:cueProgrammer とメディアエンジンの間でデータを交換するには、TCP/IP プロトコルを使います。

出来れば、固定 IP アドレスをお使いになられることを推奨致します。プライベートネットワークの IP アドレスは通常 192.168.xxx.xxx.ではじまります。

	注意：PC-mediaengine 間で同じ IP アドレスを指定することはできません。

PC とメディアエンジンの接続を調べるには、ウィンドウズプログラム PING をお使いになると便利です。

Start/Run をクリックしてエンターして下さい。

PING 192.168.123.1

まず mediaengine の IP アドレスを入力しなくてはなりません。正しいリプライタイムが表示されます。「 *time transgression requirement* 」というメッセージが表示された時には、PC とメディアエンジンの接続がきちんとなされていないということです。

サブネットマスクは通常インターネットと同じです。15PC 以下のネットワークであれば、1 つのサブネットマスクで操作されているはずです。

サブネットマスクはお使いのウィンドウズとメディアエンジンの IP アドレスがどこだけ異なっていても許されるのか定義します。仮に以下のサブネットマスクを使用しているとしましょう。

255.255.255.0

お使いのウィンドウズ PC とメディアエンジンの IP アドレスの中で、異なる部分が許されるのは、最後のポイントだけです。他の数字は同じでなくてはいけません。

もし以下のサブネットマスクをお使いでしたら：

255.255.0.0,

最後の 2 数字だけがフリーとなっています。

14.3.12. オプション／ゲート

このメニューを使うと TCP/IP プロトコルの設定が可能になります：

DNS1	プライマリードメインネームサーバーの IP アドレス
DNS2	セコンダリードメインネームサーバーの IP アドレス

14.3.13. 時計／設定

システムの時計を設定します。

14.3.14. フロッピー／ロード

ショーを 3.5 インチディスクから読み込みます。

14.3.15. フロッピー／保存

ショーを 3.5 インチディスクに保存。

14.3.16. フロッピー／アップデート

新しいソフトウェア（ファームウェア）をディバイスに読み込む。

14.3.17. Misc／アバウト

現在使用しているソフトウェア（ファームウェア）のインフォメーションとインフォメーションのステータスを表示します。

14.3.18. Misc／キー

メディアエンジンフロントパネルのオペレーションエレメンツテスト機能（キーとジョグダイアル）

14.3.19. ログイン／ログイン

ユーザーのパスワードと進入可能領域について定義と変更を行います。このログイン機能により、システムを理解していない人や悪意のアクセスから mediaengine を守ることができます。

14.3.20. ログイン／パスワード変更

システムのパスワードを変更します。

14.4. e:cue フェーダーユニットのアライメント

e:cue のフェーダーユニットは mediaengine に直接繋げることも出来ます。この場合、未定義でプログラムショーに介入出来ます。（サブマスターの変更、キューリストの停止等）

ユーザーの介入出来る範囲は、e:cue の Programmer を使ってプログラム中に設定出来ます。

14.5. e:cue キーパッドのアライメント

メディアエンジンのキーボードポートに e:cue のキーパッドを接続して使用することも出来ます。e:cue のソフトウェア、Programmer で各キーに機能を割り当てることも可能です。

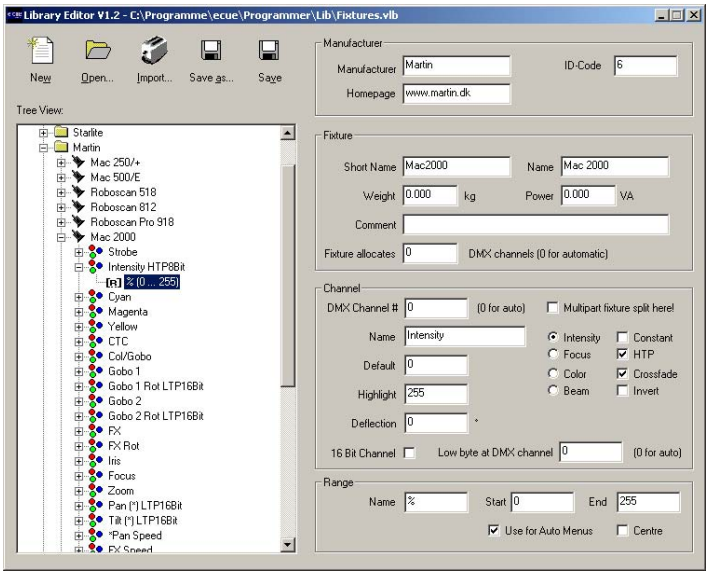
15.追記

15.1.ライブラリーエディタ

各種 DMX 器材のライブラリーは *Library Editor.*（ライブラリーエディタ）を使って変更したり、そこにない灯体情報を新たに登録することが出来ます。ショーには常に現在のライブラリー情報が保存されます。ショーが外部の PC と、他のライブラリーで読み込まれた場合、*e:cueProgrammer* はローカルライブラリーデータを使ってショーを取り込みます。

この動作は *File/Open with Lib* コマンドで変更出来ます。 この場合、ショーファイルに保存されたライブラリーデータが使用されます。この方法を使うと、オリジナル PC に保存されたショーと同じ形式でショーが再生されます。

	注意：*File/Open with Lib* 機能は機能に制限をかけます。



メーカー名、灯体の名称、チャンネル、定義値などは階層で編集出来ます。

使用可能なアクション：　アイテムの追加 (INS)、アイテムの削除 (DEL)、アイテムをクリップボードにコピー(Ctrl+C) 、クリップボードからペースト (Ctrl+V)

ディバイスのチャンネルは logical row で表示されます。使われていない DMX チャンネルをスキップするには、*"DMX channel #"* を使用して下さい （以下をご参照下さい）

15.1.1.　プロデューサーの設定

▸ Manufacturer（マニュファクチュアラー）
DMX 器材のメーカー名、サプライヤー名称

▸ ID-Code（ID コード）
器材の one-to-one ナンバー

▸ Homepage（ホームページ）
メーカー情報の参照ホームページ　　※オプション入力

15.1.2.　DMX ターミナル（灯体）の設定

▸ Short Name（ショートネーム）
ここで登録する DMX 器材は、ショートネームを施すことができます。Programmer ソフトウェアではこの名称が表示されます。*ショートネーム*は*ライブラリー*に 2 度登録することは出来ません。

▸ Name（名前）
DMX 器材の正式名称を記載します。

▸ Weight（重量）
ディバイスの重量　　※オプション入力

▸ Power（パワー）
ディバイスの消費電力　　※オプション入力

▸ Comment（コメント）
灯体に関する個別のコメント※オプション入力

（例：灯体のディップスイッチコンビネーションなど）

▸ Fixture allocates（灯体割り当て）
ディバイスに割り当てられた DMX チャンネルのナンバー設定、デフォルト 0（自動的に算出されます）

15.1.3.　チャンネルの設定

特定のチャンネルの明細

▸DMX Channel #（DMX チャンネル番号）
マニュアルチャンネルセレクション。チャンネルはトップから（1 から）カウントダウンされます。

▸Channel Name（チャンネルネーム）
チャンネルの名前。ツリービューではチャンネルは logical row に置き換えられます。各チャンネルには名前（パン、チルト、カラー等）は、ここで特定します。

▸Default（デフォルト）
チャンネルに変更が施されない限り、プリセットの値が出力されます。

▸Highlight（ハイライト）
ハイライト仕様で使われる値（7.5 をご参照下さい）

▸Deflection（偏向）
灯体の最大回転角度のためのオプションエントリー

▸Intensity / Focus / Color / Beam（照度／フォーカス／カラー／ビーム）
現在のセレクションのチャンネルグループを定義します。
Intensity channels（照度チャンネル）：
Focus channels（フォーカスチャンネル）：
（通常はパン／チルト）
Color channels（カラーチャンネル）：
Beam channels（ビームチャンネル）：

▸Constant（コンスタント）

チャンネルは修正された値に定義され、誰でもが修正を施せるわけではありません。

▸16 Bit（16 ビット）
16 ビット DMX チャンネルを定義します。最初の DMX バイトは MSB、次の DMX バイトは LSB となります。

▸HTP
キューリストが Programmer で HTP と定義されていれば、チャンネルを HTP チャンネルとして使用することが出来ます。

▸Crossfade（クロスフェード）
Programmer を使ってフェードを定義します。もしこの仕様が無効な時、フェードタイムは動作しません。

▸Invert（反転）
チャンネルのすべての出力値を反転します。

▸Multipart fixture split here

数ある DMX 器材の中には連続して DMX チャンネルが設定できないものもあります。このオプションは 2 分された DMX チャンネル群を定義します。

▸DMX チャンネルでのローバイト

16 ビットローバイトチャンネルが動作しない時には、チャンネルのロケーションはここで入力することが出来ます。チャンネルは 1 から始まります。

15.1.4. レンジ

▸Range Name（レンジネーム）

もし値が設定されていれば、名前を付けることも出来ます（例えばゴボ等）。 この値は、Programmer のコンテクストメニューで呼び出すことが出来ます。

▸Start（スタート）

値のはじまりを定義します。

▸End（終わり）

値の終わりを定義します。

▸Use for Auto Menus（自動メニューを使う）

オプション入力

▸Centre(レンジの中央点)

15.2.キーボードのコンフィギュレーション

すべての機能はキーボードの頭文字操作でより簡単に行うことが出来ます。キーの割り振りはオーバービューチャートを使ってソフトウェアに追加することも出来ます。

キーボードの現在のコンフィギュレーションは、ソフトウェアのヘルプユティリティーをご参照下さい。F1 でオンラインサポートに接続することが出来ます。

15.3.デュアルヘッド

このソフトウェアはモニターを 2 つ使って操作することをお薦めします。Add-on to ウィンドウを使ってセカンドモニターを使用するに設定して下さい。

プログラムのメインビューから DMX アウトプット、ジョグダイアル、キューリストコンテント、フローシートなどをセカンドモニター側に表示することができます。

同一 PC で WYSIWYG を操作すると、片方のスクリーンでは WISYWYG のレンダリング画面を、もう 1 つのスクリーンでは programmer ビューを表示することが可能です。他の PC をご用意していただく必要はありません）。Programmer は変更されたコンフィギュレーションを自動的に変更しますので、標準の PC で十分です。

この機能を有効にするには、まずお手持ちの PC にデュアルヘッド対応のグラフィックカードを挿入し、各種設定を終わらせておく必要があります。

動作確認の出来ているデュアルヘッドグラフィックカードは以下をご参照下さい：

- ATI Radeon 7500

- ATI Radeon 8500

16.テクニカルデータ

16.1.e:cueProgrammer ソフトウェア

16.1.1. 概要

必要最小構成(PC)

- ペンティアム 2CPU を内蔵した PC/AT 互換機（PC9800 シリーズを除く）
  64MB RAM 必須
  OS: Microsoft Windows 98SE/2000/NT/ME/XP ベースで動作(2002 年 10 月現在)

主な機能

- 非常に簡単に操作できるソフトパッチ部
- DMX コントロールライティングディバイスをフルサポート。
- 最大 4096DMX チャンネルをコントロール。
- マクロ機能搭載。
- 著明メーカーの灯体を網羅したスタンダードライブラリー
- 1 ディバイスで複数の灯体グループを一元化してコントロール
- ユーザー自身でオリジナル灯体データが作成できるライブラリエディターをバンドル
- 複数灯体のグルーピング機能
- パッチデータをはじめとするさまざまな情報を CSV 形式でエクスポート可能
- プログラミングセクション
- Programmer、キューリストコンテンツおよび DMX 出力の 3 つのビューを切替表示
- サークルやウェーブなどを簡単に作成できる機能豊かなエフェクトエンジン
- ハイライト／ブラインド／フリーズ機能搭載

・カラーピッカーによるスピーディーな色決め（ウォッシュ使用時）が可能。

・キューフェードにリニア／ソフトスタート／ソフトストップ／ソフトモーションを設定可能。
また、任意のグループや列の中央／左右から徐々にスタートできるフェードディレイ機能を追加。

・通常のキュー進行に加え、外部タイムコード／MIDI／ビートなどからのトリガーも可能。

・マクロ機能搭載。各キューもしくはキューリストに最大５つまでのアクション（音を出したり画像を出したり他のキューに連動させたり etc...）を登録可能。また、マクロはユーザー自身でプログラミングすることができ、C 言語を使ってさらに複雑なマクロも構成することが可能。（マクロ作成オプションも用意しています）

・チェイススピード学習機能

・テキストファイルに既存のショーデータ詳細をエクスポート可能

PROGRAMMING（プログラミング）

- ３つのビューモード：*Programmer ビュー*、*キューリストコンテント*、*アウトプットビュー*
- エフェクトを素早く作成することの出来るライブ FX エンジン（サークルドライブ、ウェーブ、チェイサー等）
- *Highlight（ハイライト）*、 *Blind（ブラインド）*、*Freeze（フリーズ）*
- 灯体のライティングとキューリストに *HTP*、*LTP* を同時使用
- *カラーピッカー*
- 値の簡単設定
- *シーンをリニア*、*ソフトスタート*、*ソフトストップ*、*ソフトモード*他でフェード可能
- 非同時性のフェード（まず左、まず右、他）時間設定、サブグループ自由
- キューリストのループを組み合わせる
- マクロを使って５つまでの任意のアクションの組み合わせが可能
- オートマチックミューチュアルはキューリストを除きます
- キューリストの任意の番号を同時に有効に出来ます
- コマンドバー
- *チェイサースピードティーチイン*：チェイサーの速度自動設定

- ショーをスクリプトとしてエキスポート
- 2 モニターの使用をサポート

WIZARDS（ウィザード）
- チェイサーウィザード： 自動チェイサー
- チェックイットウィザード： ライブラリーデータを使って、ディバイスのテストサンプルを自動作成
- スクロールテキストウィザード： スクロールテキストを簡単作成

OUTPUT DEVICE（出力ディバイス）
- ナノ、トワイライト、イーノード 512、イーノード 1024 を使って DMX 出力
- *メディアエンジン (e:cue メディアエンジンのプログラミングのための特別仕様)*
- *トリガーマシーン*： アクションとイベントのコンビネーション
- *Snippet Editor:* C 言語でのマクロプログラム可能
- *日の出タイマー*： 日の出／日の入時間を場所に従って自動算出
- *オンラインアクセス*：*e:cue メディアエンジン*のライブアクセス
- ショーを*メディアエンジン*に直接 *Upload（アップロード） / Download（ダウンロード）*

MORE（他）
- *ナノ、イーノード 512、イーノード 1024* で PC がなくてもショーをプレイバック
- *e:cue フェーダーユニット*対応
- *CAST 社の WYSIWYG* ソフトウェアに対応
- Programmer ソフトウェアのフリーダウンロード→ *www.ecue.tv*

*ナノとイーノード*は、メディアエンジンと違い、*Programmer* のすべての機能に対応しているわけではありません（テクニカルデータをご参照下さい）。

16.2. フェーダーユニット

16.2.1. 概要

- PC／ノートブック／メディアエンジンに RS232 インターフェイスを使って接続します
- オプション：MIDI 出力 （ドライブを外部ソフトウェア／ハードウェアに変換出来ます）
- 13 フェーダー（グランドマスター×1、サブマスター×12）
- ジョグダイアル×5
- 73 キー（内 10 キーは任意で設定出来ます）
- 本体下部に安定性のある足がついています
- 外部パワーサプライユニットを使用しての電源供給
- 箱型アルミ製フロントパネル
- 重量： 約 1.5kg

16.2.2. RS232 インターフェイス

9 ピン Sub-D オス

Pin	機能
1	*n.c.*
2	TxD
3	RxD
4	*n.c.*
5	GND
6..9	*n.c.*

16.2.3. Midi インターフェイス

5 ピン DIN ジャック

Pin	機能
1	*n.c.*
2	*n.c.*
3	*n.c.*
4	Data –ve
5	Data +ve

16.2.4. パワー

2.1mm プラグ、オス

Pin	機能
1	AC in (7.5-15 V)
2	AC in (7.5-15 V)

16.3.メディアエンジン

16.3.1. 概要

The media engine disposes beside two DMX outputs of numerous interfaces which enable to combine the light show with other devices resp. generically with external events.

HARDWARE（ハードウェア）
- *e:cue* 生産の PC、*e:cue Linux* オペレーティングシステム
- イルミネーションを施した LC ディスプレイ／6 キー＋ジョグダイアル
- 3.5 インチフロッピードライブ
- EIA 規格 2U ラックマウント (483x88x350 mm)
- 重量 約 6.5kg

INTERFACES（インターフェイス）
- DMX （2 ライン 1024 DMX チャンネル）
- e:node512 もしくは e:node1024 を使って 8DMX ライン（4096 チャンネル）までの使用が可能
- イーサネット
- VGA
- オーディオ出力
- MIDI イン、MIDI アウト、MIDI スルー
- SMPTE イン
- 6 インプット 10..24V
- 2 アウトプット +12V, 25mA
- キーボード
- シリアル RS232×2

FEATURES（仕様）
- e:cueProgrammer のリプライユニット

- イベントコントロールレンディションのためのトリガーマシン機能
- 日の出タイマー （天文時計）
- JPG インジケーション
- MP3 再生
- SMPTE と MIDI タイムコード（MTC）を受領
- WLAN（ワイヤレス LAN）を使って PC／ノートブックとワイヤレスで接続可能
- e:cue フェーダーユニットを使って、自動再生の反転が可能
- シリアルインターフェイスの任意設定
- 外部 LCD ターミナル対応（e:com／イーコム）
- 詳しいステータス表示
- DMX テスター
- Midi モニター
- FTP サーバー
- パスワード保護
- フリーファームウェアのアップデート

16.3.2. メディアエンジンインターフェイス

インターフェイス	プラグ	機能
DMX×2	5 ピン XLR プラグ	DMX512 出力
I/O インターフェイス	6 x In, 2 x Out	Potential-free entries (10..24V), galvanic separated outlets (+12V, 50mA)
イーサネット	RJ45	TCP/IP メディアエンジンサーバー HTP サーバー FTP サーバー ACN ready
RS232×2	9 ピン Sub-D	外部メディア再生ディバイスのアライメント（CD、MD、DVD 他）
オーディオ		MP3 オーディオプレイバック
VGA	標準 VGA	1024 x 768 ピクセル、65,000 カラー JPEG レンダリング&モニタリング機能
USB×2	USB 1.0	
キーボード	PS/2	MF102 10 キー キーボード／e:cue キーパッド
マウス	PS/2	
MIDI	DIN （イン、アウト、スルー）	Midi – 送受
SMPTE		SMPTE タイムコード – 送受

16.3.3. オプション

ワイヤレス LAN	ワイヤレスネットワーク。ノート PC を使って、ワイヤレスでメディアエンジンにアクセス可能
フェーダーユニット	特定のキューを直接再生。サブマスター／グランドマスター／機能キーにアクセス。
ハードディスク	20 GB ハードディスク

オプションについてのより詳しいインフォメーションに関しては e:cue のホームページをご覧いただくか、お近くのディーラーまでご相談下さい。

16.3.4. SMPTE インターフェイス （6.5mm ソケット）

Pin	機能
1	Common
2	Neg
3	Pos

16.3.5. Midi インターフェイス （5 ピン DIN プラグ）

Pin	機能
1	*n.c.*
2	*n.c.*
3	*n.c.*
4	Data –ve
5	Data +ve

16.3.6. DMX インターフェイス （5 ピン XLR プラグ、メス）

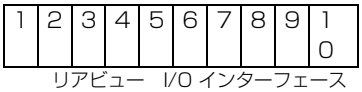
Pin	機能
1	Digital Ground
2	Data –ve
3	Data +ve
4	*n.c.*
5	*n.c.*

16.3.7. RS232 インターフェイス （9 ピン Sub-D プラグ、オス)

Pin	機能
1	*n.c.*
2	RxD
3	TXD
4	*n.c.*
5	GND
6..9	*n.c.*

16.3.8. SPS インターフェイス(Phoenix ソケット、10 ピン、オス)

Pin	機能
1	+12 V DC out
2	GND
3	In 1 (5..25V DC)
4	In 2 (5..25V DC)
5	In 3 (5..25V DC)
6	In 4 (5..25V DC)
7	In 5 (5..25V DC)
8	In 6 (5..25V DC)
9	Out 1（オープンコレクター、最大 20mA/24V）
10	Out 2（オープンコレクター、最大 20mA/24V）



リアビュー I/O インターフェース

16.3.9. オーディオ出力 （3 ピン XLR プラグ、オス）

Pin	機能
1	Digital Ground
2	オーディオ出力（右）
3	オーディオ出力（左）

16.4.Nano

概要

- DMX512 ／ RS232 インターフェイス
- LED ステータス×2
- *e:cueProgrammer でプログラム可能*
- ライブモード 256 DMX チャンネル、
  スタンドアローンモード 32DMX チャンネル
- 最大 255 キューまで使用可能なキューリスト（保証出来るのは 64 キューまでです）
- リニアフェード、時間設定自由
- 非同時性フェード（内側のライト灯体をまず初めに、外側を初めに、右側を初めに）、時間設定自由
- スタンドアローンモードでは fx エンジンは使用出来ません
- 外部パワーサプライユニットでの電力供給
- アルミ製、防水加工（105×57×32）
- オプション： ウォールホルダー
- 重量： 約 500g

16.4.1. RS232（9 ピン Sub-D プラグ、オス）

Pin	機能
1	*n.c.*
2	TxD
3	RxD
4	*n.c.*
5	GND
6..9	*n.c.*

16.4.2. パワー（2.1mm プラグ、オス）

Pin	機能
1	AC in (7.5-15 V)
2	AC in (7.5-15 V)

16.4.3. DMX インターフェイス （5 ピン XLR プラグ、メス）

Pin	機能
1	Digital Ground

2	Data –ve
3	Data +ve
4	*n.c.*
5	*n.c.*

16.5. Twilight

16.5.1. 概要

- USB インターフェイス
- *e:cueProgrammer* を使用してプログラミング
- 一度に 8 トワイライトまで接続可能
- 512DMX チャンネル／DMX512 出力
- 5 ピン XLR プラグ
- USB パワーサプライ／パワーサプライユニットの必要はありません
- アルミ製、防水加工（155×57×32）
- 重量： 約 700g

16.5.2. DMX インターフェイス （5 ピン XLR プラグ、メス）

Pin	機能
1	Digital Ground
2	Data –ve
3	Data +ve
4	*n.c.*
5	*n.c.*

16.5.3. USB

USB 1.0 対応

17.サポート

当社では、e:cue に秘められた数々のポテンシャリティをユーザーの皆様により深く知っていただき、これからのライティングデザインに役立てていただけるようさまざまなサービスをご提供しています。

全てのサポートを可能にするには、まずお手元のレジストレーションカード（保証登録カード）に御記入頂き、当社にてお客さまの情報を登録する必要があります。
登録された情報はユーザーに対するより豊かなサービス活動に役立てられるばかりでなく、故障や不具合時の迅速なサポートにも貢献します。ぜひこの機会に御登録頂きますようお願いします。ただし、一旦登録された情報は当社の承諾無しに譲渡できませんので予めご了承ください。

また、e:cue 製品紹介サイト
www.soundfarm.co.jp/ecue

また、製品応用事例やたびたび寄せられる Q&A、さらに研究／開発者とユーザーを結ぶ BBS フォーラムなどを御用意しています。必要に応じて御参照頂いたり、他のユーザーとのコミュニケーションなどにお役立てください。

ドイツ Beiner van Beeck 社の E:cue オフィシャルサイトでも、製品に関する最新の情報やアップデータをいつでも入手することが出来ます。ここでは世界のさまざまな開発者が研究事例を発表したり、プロジェクトに関するフォーラムを開催しています。詳しくは

http://www.ecue.tv(英語)
をご覧ください。

18.Masthead, Copyright

© 2002 e:cue Beiner van Beeck GbR, Arminiusstraße 46, D-33100 Paderborn.
www.ecue.tv. All rights reserved.

e:cue は Beiner van Beeck 社の登録商標です。WYSIWYG はカナダ、トロントを主たる所在地とする CAST ライティングリミテッド社の登録商標です。他のすべての会社名や製品名、登録商標は各国の法律で保護されています。
e:cue プロパティやキャパシティ、テクニカルデータはお使いのオペレーティングシステムによっても変わってきます。またハードウェアによっても変わってきます。
e:cue 製品はその外観や製品仕様について予告することなく変更する場合があります。
e:cue 製品は、日本、韓国、中国、台湾、インドネシア、シンガポール、タイ、フィリピン、ミャンマーに対し、次のディストリビューターによりサポートされます。
ディストリビューター
〒7010131
岡山市花尻みどり町 3-112
有限会社　サウンドファーム　　086-214-0852
また、製品の保証は別紙製品保証書の記述に基づいて行われます。

19.キーワード索引

16 Bit 99
Action 47, 54
Application Setup 67
Auto Menus 99
Auto release 56
Automatisierung 58
Blind 41
CD Player 25
Centre 99
Channel Name 98
Chase Speed 56
Chaser Wizard 48
Clustersize 64
Color Picker 42
Constant 99
Crossfade 99
Cuelist content 36
Cueliste 46
Daily 69
Dark 23
Decrease 23
Default 98
Deflection 98
Delay 47, 50
Device Mapper 26
DMX Adress 31
Dualhead 100
Edit 52
Effekt-Generator 43
Fade 53
Fade In 47
Faderunit 88
File Setup 64
Fixture Type 32
Fixtures 32
FTP Access 24
FX Preset 41
Ger teoptionen 33
Group 39
Highlight 41
Highlight Dimm 64
Hourly 69
HTP 55, 99
ID# 31
Increase 23
Invert Pan 33
Keyboard 69
Keypad 96
Label 69
Library Editor 97
Limit 33
live fx 43
Loop 53
Manual 54
Media engine 90
Media Play 70
Media Stop 70
MIDI Control Change 70
MIDI Message 70
Midi Note 69
Nano 28
Navigator 35
Offset 43
Open 20
Park 41
Pause 46
Play 46
Power down 70
Power Up 69
Prehea 33
Programmer view 36
Range Name 99
Ratio 43
Record Cue 44
Release time 57
Renumber 52
Repeat 47
RS232 70

Save 20
Screen Name 31
Set Output up 70
Size 43
SMPTE 70
Speed 43
SPS Output 70
Stage output 37
Statusbar 25
Stop 46
Swap Axis 33
Tastaturbelegung 100
Technische Daten 101
Timecode 57
Timing 47
Toolbar 25
Touch 40, 64
Trigger-Machine 68
Twilight 29
Universe 31
Untouch 40
Wait 54
Wysiwyg 60